Visualizing media with interactive multiplex networks

Haolin Ren

université de BORDEAUX

École doctorale 39 : Mathématiques et Informatique

Doctorat Université de Bordeaux

THÈSE

pour obtenir le grade de docteur délivré par

Université de Bordeaux

Spécialité doctorale “Informatique”

présentée et soutenue publiquement par

Haolin Ren

le 14 mars 2019

Visualizing media with interactive multiplex networks

Directeur de thèse : **Guy Melançon**
Co-encadrant de thèse : **Marie-Luce Viaud**

Jury

Mme Florence Sedes,	Professeur, Université Paul Sabatier	Rapporteur
M. Gilles Venturini,	Professeur, Université de Tours	Rapporteur
M. Jean Carrive,	DR., Institut national audiovisuel	Président
M. Gayo Diallo,	Université de Bordeaux	Examinateur
M. Benjamin Renoust,	Université de Osaka, JP	Examinateur

Université de Bordeaux
UMR CNRS 5800 LaBRI, F-33405 Talence, France

Cartographier les médias avec des réseaux multiplexes interactifs

Les flux d'information suivent aujourd'hui des chemins complexes: la propagation des informations, impliquant éditeurs on-line, chaînes d'information en continu et réseaux sociaux, emprunte alors des chemins croisés, susceptibles d'agir sur le contenu et sa perception. Ce projet de thèse étudie l'adaptation des mesures de graphes classiques aux graphes multiplexes en relation avec le domaine étudié, propose de construire des visualisations à partir de plusieurs représentations graphiques des réseaux, et de les combiner (visualisations multi-vues synchronisées, représentations hybrides, etc.). L'accent est mis sur les modes d'interaction permettant de prendre en compte l'aspect multiplexe (multicouche) des réseaux. Ces représentations et manipulations interactives s'appuient aussi sur le calcul d'indicateurs propres aux réseaux multiplexes.

Ce travail est basé sur deux jeux de données principaux: l'un est une archive de 12 ans de l'émission japonaise publique quotidienne NHK News 7, de 2001 à 2013. L'autre recense les participants aux émissions de télévision/radio françaises entre 2010 et 2015. Deux systèmes de visualisation s'appuiyant sur une interface Web ont été développés pour analyser des réseaux multiplexes, que nous appelons «Visual Cloud» et «Laputa».

Dans le Visual Cloud, nous définissons formellement une notion de similitude entre les concepts et les groupes de concepts que nous nommons *possibilité de co-occurrence* (CP). Conformément à cette définition, nous proposons un algorithme de classification hiérarchique. Nous regroupons les couches dans le réseau multiplexe de documents, et intégrons cette hiérarchie dans un nuage de mots interactif. Nous améliorons les algorithmes traditionnels de disposition de mise en forme de nuages de mots de sorte à préserver les contraintes sur la hiérarchie de concepts.

Le système Laputa est destiné à l'analyse complexe de réseaux temporels denses et multidimensionnels. Pour ce faire, il associe un graphe à une segmentation. La segmentation par communauté, par attribut, ou encore par tranche temporelle, forme des vues de ce graphe. Afin d'associer ces vues avec le tout global, nous utilisons des

diagrammes de Sankey pour révéler l'évolution des communautés (diagrammes que nous avons augmentés avec un zoom sémantique).

Cette thèse nous permet ainsi de parcourir trois aspects (3V) des plus intéressants de la donnée et du BigData appliqués aux archives multimédia: Le *Volume* de nos données dans l'immensité des archives, nous atteignons des ordres de grandeurs qui ne sont pas praticables pour la visualisation et l'exploitation des liens. La *Vélocité* à cause de la nature temporelle de nos données (par définition). La *Variété* qui est un corollaire de la richesse des données multimédia et de tout ce que l'on peut souhaiter vouloir y investiguer. Ce que l'on peut retenir de cette thèse c'est que la traduction de ces trois défis a pris dans tous les cas une réponse sous la forme d'une analyse de réseaux multiplexes. Nous retrouvons toujours ces structures au cœur de notre travail, que ce soit de manière plus discrète dans les critères pour filtrer les arêtes par l'algorithme Simmelian backbone, que ce soit par la superposition de tranches temporelles, ou bien que ce soit beaucoup plus directement dans la combinaison d'indices sémantiques visuels et textuels pour laquelle nous extrayons les hiérarchies permettant notre visualisation.

Mots Clés: Réseau Multiplexe, Graphe Dynamique, Graphe Temporel, Détection de Communauté, Visualisation, Big Data, Analyse Visuelle

Laboratoire:
Laboratoire Bordelais de Recherche en Informatique (UMR 5800)
351, cours de la Libération, 33405 Talence cedex, France

Visualizing media with interactive multiplex networks

Nowadays, information follows complex paths: information propagation involving on-line editors, 24-hour news providers and social medias following entangled paths acting on information content and perception. This thesis studies the adaptation of classical graph measurements to multiplex graphs, to build visualizations from several graphical representations of the networks, and to combine them (synchronized multi-view visualizations, hybrid representations, etc.). Emphasis is placed on the modes of interaction allowing to take in hand the multiplex nature (multilayer) of the networks. These representations and interactive manipulations are also based on the calculation of indicators specific to multiplex networks.

The work is based on two main datasets: one is a 12-year archive of the Japanese public daily broadcast NHK News 7, from 2001 to 2013. Another lists the participants in the French TV/radio shows between 2010 and 2015.

Two visualization systems based on a Web interface have been developed for multiplex network analysis, which we call "Visual Cloud" and "Laputa". In the Visual Cloud, we formally define a notion of similarity between concepts and groups of concepts that we call co-occurrence possibility (CP). According to this definition, we propose a hierarchical classification algorithm. We aggregate the layers in a multiplex network of documents, and integrate that hierarchy into an interactive word cloud. Here we improve the traditional word cloud layout algorithms so as to preserve the constraints on the concept hierarchy. The Laputa system is intended for the complex analysis of dense and multidimensional temporal networks. To do this, it associates a graph with a segmentation. The segmentation by communities, by attributes, or by time slices, forms views of this graph. In order to associate these views with the global whole, we use Sankey diagrams to reveal the evolution of the communities (diagrams that we have increased with a semantic zoom).

This thesis allows us to browse three aspects of the most interesting aspects of the data miming and BigData applied to multimedia archives: The *Volume* since our archives are immense and reach orders of magnitude that are usually not practicable for the visualization; *Velocity*, because of the temporal nature of our data (by definition). The *Variety* that is a corollary of the richness of multimedia data and of all that one may wish to want to investigate. What we can retain from this thesis is that we met each of these three challenges by taking an answer in the form of a multiplex network analysis. These structures are always at the heart of our work, whether in the criteria for filtering edges using the Simmelian backbone algorithm, or in the superposition of time slices in the complex networks, or much more directly in the combinations of visual and textual semantic indices for which we extract hierarchies allowing our visualization.

Keywords: Multiplex Network, Dynamic Graph, Temporal Graph, Community Detection, Visualization, Big Data, Interaction, Visual Analytics

Laboratory:
Laboratoire Bordelais de Recherche en Informatique (UMR 5800)
351, cours de la Libération, 33405 Talence cedex, France

Table of contents

Chapter 1

Introduction

This thesis was funded by ANRT through a collaboration involving INA (Institut National Audiovisuel, Paris, FR) and LaBRI (Laboratoire Bordelais de Recherche en Informatique, UMR CNRS 5800, Talence, FR).

INA is a public institution archiving audiovisual productions, producing and publishing audiovisual and multimedia content for all audiences. INA is also a training and research center that aims to develop and transmit knowledge in the audiovisual, media and digital fields. The thesis took place as part of activities of INA *Recherche & Innovation Recherche & Innovation* is a research team led by Marie-Luce Viaud focusing on information retrieval and visualization interfaces.

This work was also accomplished in co-supervision by Benjamin Renoust from Osaka University (formerly at NII in Tokyo), in particular the part focusing on video document visualization.

This section briefly summarizes the work and results of the main research during the doctoral period. The manuscript is organized into two main parts, one focusing on the visualization of video documents content, the other looking at how dynamic network analysis can support the understanding of media as a social ecosystem.

1.1 Video documents content

The size of digital news archives makes it necessary for media studies to rely on automatic processing for quantitative analysis, not limited to their textual content. Visual analytics of multimedia data proposes then to extract high-level representations and support high-end analysis of multimedia concepts (not limited to text). Advances in computer vision now allows the extraction of visual semantic concepts, which can be used in turn to index video documents in an archive [32]. Querying and

retrieving relevant information still remains a difficult task, one with a relatively high cognitive cost for users, especially considering semantic ambiguity. This has stimulated information visualization to support the strategies adopted by users to find their way in the information space formed by results. In particular, tag clouds can give an overview of this semantic space and support exploratory tasks [66].

With news videos, the heterogeneous combination of both textual and visual concepts presents an interesting challenge. Modeling query results as multiplex document networks – each semantic concept being associated to a layer [54] – can turn this heterogeneity as an advantage. [53] had shown how a Layer Interaction Network (hereafter *LIN*) derived from the original multilayer network enables the design of advanced interaction and coordination, and thus eases exploration.

1.2 French medias as a social ecosystem

The social and political life of a community is partly shaped by the voice of different actors and groups of actors is echoed in the media. In the context of INA's OTMedia project[1], different measures and visualizations supporting the analysis of medias visibility were developed (newspapers, radio, TV and even social medias).

We have investigated radio and TV programs broadcasted between 2011 and 2015, and have more particularly looked at co-invitation patterns hoping to locate different communities of actors. By doing so, we hoped to answer simple questions such as "Are there groups of person being co-invited often with one another?" "Can we provide an overview of all co-invitations, groups and how they evolve in time?", etc. Because co-invitations spread over a five year period, the notion of a community had to be tackled in a manner consistent with the fact that people come and go between groups.

The data we use naturally comes as a network of people being connected as they are co-invited on TV or radio shows. The avenue we borrow to provide answers to the previous questions is to "visualize" the data and provide insight on its "structure" and display graphical patterns that the human eye is extremely well trained to detect [75] [76]. The structure of data is intimately linked to the notion of a "group": a set of elements showing relative homogeneity [68, 62]. Now, the data we use is modeled as a graph where nodes correspond to people and links correspond to co-invitation to a same TV or radio show. For graphs, homogeneity is generally referred to as *modularity* and most often measured in terms of internal connectivity versus external connectivity. A popular modularity measure was introduced by Newman [46].

[1] See www.otmedia.fr

Dynamic graphs – those where nodes and edges vary in time – raise quite a number of issues. Properly defining the notion of a group or community in a dynamic graph is difficult as shows the lack of consensus in the literature [57]. Their visualization is another difficult and interesting problem [11]. The challenge indeed is to be able to display changes through a graphical representations capable of supporting the user's mental map [4].

As if the challenge was not hard enough, our aim is to help users track changes in the community structure of a network – which often are termed *dynamic communities* [57]. This notion however hides an implicit assumption, that communities remains relatively stable over time. This "stability" assumption simply does not hold in the context of our work, as we shall see.

We study the French medias, and look more particularly at how people (politicians, actors, sportsmen, commentators, etc.) get co-invited on the same radio or TV shows over the 2010 – 2016 period. Being able to track and understand patterns potentially sheds light on editorial policies and potential biases induced from repeated co-invitations on given topics.

Our contribution is a fully equipped exploratory dashboard allowing users to select and track groups of people over time in the "co-invitation network". Communities, either suggested using standard algorithms or deliberately selected by users, can then be examined from different angles, and through different signals reflecting the level of "activity", "stability" or "renewal" rate of a group.

1.3 Overview of main contributions

In this thesis we focus on data modeling, visualization and analysis approaches toward a better understanding of complex network and time evolving network. Information visualization is the primary focus of our investigations. The four main contributions are as follows:

Temporarily and heterogeneity in graph communities from media archives

- Heterogeneity within the document context. Documents have many properties such as channels, broadcasts, type of broadcast etc.
- Heterogeneity within the document itself: mining the keywords, computer vision cues, face detection.

Ways to model the heterogeneity

- heterogeneity through multiplex networks
- heterogeneity through bipartism

The first contribution of our design study is a formal description using Brehmer *et al.*'s approach [14] of the tasks we need to support. Each task corresponds to a domain question and a set of operations conducted on the relevant data. Based on these task descriptions, our second contribution is the design of a visual analytics dashboard shown in Fig. 4.5, consisting in:

- A dual community detection approach based on two complementary perspectives: first on the overall link structure with time-independent groups confronted to time-interval induced groups;
- A set of statistics turned into indicators of time-dependent community cohesion;
- A system combining and synchronizing several representations of communities, together with dedicated interactions.

Our dashboard uses Bobo Nick's [48] Simmelian backbone filtering approach to downsize the data and provide a fluid navigation of our data. The curves of time-dependent statistics can be used to filter data according to time. The iterative design of the dashboard was guided by close consultation with two expert users. Our last contribution is two use cases that showcase how the dashboard supports the analysis work-flow.

1.3.1 Visually exploration of network over time

We design and produce a visual framework to represent temporal graphs, to detect communities and to show their evolution over time.

This work is done in the French audiovisual institute and the constraints are strong because the framework produced should handle real data, is tested and used by real expert users, to produce real studies at real scale. Then, the framework should contain several modules to ensure it use in real environment:

- a module to build a temporal graphs from tabular data or given graph models
- a module to observe and evaluate the temporal distribution of the data
- a visual framework to visualize the graph and its evolutions over time and allows users to manipulate the data.

1.3.2 A framework to interactively analyze the labeled Videos

We build a multi-layered visual cloud for semantic concepts visualization. This visual cloud has four features:

- Hierarchical placement. After modifying the traditional tag cloud location calculation algorithm, our visual cloud implements a mix of tags and snippets, and the placement of each concept is hierarchically distributed.

- Network behind. Concepts of visual cloud are from interaction network, so the connection between concepts is preserved and users can interact to understand the connections between concepts.

- Multiple interaction. A time line is designed to filter the time varying visual cloud. Click, mouse hover, double click etc. several interactive mode is designed in our application for user explore the data.

- Heat-map highlight. A heap-map is creatively designed in our visual cloud for highlight the interest region of concepts, helps user quickly have an overview of all concepts.

1.3.3 Research and propose some algorithms

In this thesis, we studied the cluster algorithm and proposed our own methods for better understand the graph.

- A hierarchical cluster for complex graph.

- Several indexes for analyzing the communities' evolution over time.

- In-depth study of the Simmelian backbone algorithm and dynamic graph clustering algorithm.

Chapter 2

State of Art

2.1 About clustering

There are two different clustering method: Clustering via inter-document similarity. The best-known and earliest research on document clustering for search user interface is the Scatter/Gather project [50], documents are clustered into topically-coherent groups, and presenting descriptive textual summaries to the user. The usability study showed that the use of Scatter/Gather on a large text collection successfully conveyed some of the content and structure of the corpus. Usability study results suggest that user dislike organizations that show inconsistent levels of description [21].

Clustering according to the shared common term. Monothetic clustering algorithms [13] build clusters around dominant phrases, which give rise to more understandable labels. An analysis of the queries for which clusters were selected suggested that they are helpful primarily for moving documents that are low in the standard search rankings up higher. This happened on those occasions in which the query was ambiguous and the primary sense was not shown near the top of the search results, or when the query was specified very generally [31].

A primary problem with clusters is that their contents can be difficult to understand. The study by Kleiboemer et al. [35] found that for non-expert users the results of clustering were difficult to use, and that graphical depictions were much harder to use than textual representations, because documents' contents are difficult to discern without actually reading some text. The human perceptual system is highly attuned to images, and visual representations can communicate some kinds of information more rapidly and effectively than text [65].

A notable successful use of visual cues in search interfaces is color highlighting of query terms in documents, and bolding of query terms in document summaries in retrieval results [41].

2.2 Detecting communities in dynamic networks

A common definition to describe dynamic networks is to consider a finite sequence of graphs $G_0, G_1, \ldots, G_k$ where the node and edge sets of graphs $G_t = (V_t, E_t)$ vary in time, and where each graph G_t is associated with a timestamp (or time interval). It is usual to refer to each of the graphs G_t as begin *static*, where G_t represents the state of the network at timestamp t.

There roughly are three approaches to address community detection in dynamic networks. They are:

- A first approach is to aggregate all graphs G_t into a single graph and apply community detection methods for static graphs. Typically, edges if the resulting graphs can be weighted according to how frequently nodes get connected through time [6] [69].
- Another set of approaches is to compute communities for each of the graph G_t.
  - Communities can then be reconciled [20] [39], simultaneously optimizing the quality of the community structure at $t + 1$ and the maximization of the likeliness with communities at t.
  - Alternatively, communities for G_{t+1} can be computed using communities for G_t as a starting point [67] [19], focusing on changes that occurred from G_t to G_{t+1}. Changes are dealt with in different manners depending on their types (community growth or contraction, community merge, split, birth, death and even resurgence; see [18]). A foreseen advantage of this approach is to compute communities for G_{t+1} in less time, assuming snapshots G_t and G_{t+1} are topologically close to each other.

As for the latter optimization approaches one has to be cautious and take into consideration the number of snapshots G_t, and the complexity of the networks at hand – that is, the implicit assumption on link persistence may not be at work. This is an important aspect in our work as our data is highly dynamic and links are far from being persistent. Also, these methods most of the time being non-deterministic they

tend to produce unstable results [58]. Hence it appears that providing users with means to inspect community structures in a sense is mandatory.

2.3 Visualizing evolving communities

Animating graphical representations is a traditional approach to indeed represent dynamic network [24] [11]. The complexity of the network, and consequently the readability of the representation may however call for alternative approaches [56]. Small multiples [25] consists in juxtaposing similar representations of varying but comparable datasets and effectively support the detection of changes in dynamic networks. This approach applies to dynamic networks but is subject to an obvious scalability issue [4]. Scalability can be improved by grouping timestamps into hierarchy that can be developed on demand as in [7].

Of particular importance here is the visualization of group structure in dynamic graphs [71].

Interactive Sankey diagrams [55] have proved to be quite useful in visually displaying evolving community structures [70]. As we shall see, Sankey diagrams can be enriched with interactions, and usefully synchronized with node-link views to support the visual inspection of evolving communities.

Chapter 3

Capturing data heterogeneity through Visual Clouds

3.1 Introduction

RECENT advances in Computer Science have brought quantitative capabilities for news analysis on a large scale, to the benefit of media studies and sociology. Beyond topic detection and tracking from the text data, the analysis of video content itself matters, due to the impact images to the viewers [9].

Visual analytics of multimedia data proposes then to extract high-level representations and support high-end analysis of multimedia concepts (not limited to text). The analysis part of multimedia analytics often requires semantic annotations describing a multimedia document or at least a part of it. Semantic concepts required for analysis are often extracted from textual annotations, when available [38]. Advance in computer vision now allows the extraction of visual semantic concepts, which can be used in turn to index video documents in an archive [32]. Querying and retrieving relevant information still remains a difficult task, one with a relatively high cognitive cost for users, especially considering semantic ambiguity. While relevance ranking (such as page ranking [16, 8]) is a powerful approach to extract a set of interesting pages, studies have shown that users usually focus on the first few pages of results.

Semantic ambiguity remains challenging the information retrieval workflow, from extraction down to restitution to users [45, 37]. This has stimulated information visualization to support the strategies adopted by users to find their way in the information space formed by results. This has stimulated the study of strategies adopted by users to find their way into the information space [42] and information visualization soon responded to tackle this task [2, 78]. In particular, tag clouds can

give an overview of this semantic space and support exploratory tasks [66]. The visual inspection of results and even snippets for the first few most relevant results cannot allow users to build a proper mental map of this space. We want to address this problem by giving access to the semantic coverage of results of a user query through proper visual representation and interaction, in a form of a multimedia analytics system delivering a visual map of the query results.

Multiplex networks are heterogeneous and complex data structures: they present interacting entities across multiple layers of interactions. One successful strategy to visualize these complex structures is to exploit an additional network: This layer interaction network describes how layers overlap and may be used to investigate the original network. When the layers correspond to semantic units in documents, the new network shows some sense of semantic hierarchy reflecting the semantic context formed by the documents. We first propose an algorithm to extract hierarchical clusters of layers in a multiplex network of documents, and then to embed this hierarchy in an interactive word cloud.

3.2 Data preprocessing

With news videos, we also challenged by another issue is raised by the heterogeneity of multimedia information, in which we must visually represent. The challenge is then to visually represent both textual and visual semantic concepts combined. Modeling query results as multiplex document networks – each semantic concept being associated to a layer [54] – can turn this heterogeneity as an advantage and derive a Layer Interaction Network (hereafter *LIN*), which offers advance interaction and coordination [53].

To this end, we propose to model query results in a multiplex network [34] since it aims at capturing multiple types of relationships [17]. as one could observe in the real world phenomenon (*e.g. friend*, *colleague*, *family* or *sports club* for social relationships). Each layer can be associated to a semantic concept shared between different documents when modeling a document network [54]. These multiplex networks of documents often show a high number of layers (as opposed to traditional multilayer networks that often consider largely below a dozen). The number of layers can been turned into an advantage by using them to form a second network, the Layer Interaction Network (hereafter *LIN*), which offers interesting opportunity in terms of interaction and coordination [53].

Detangler [53] used it to coordinate exploration in association with a *flattened* multiplex network (corresponding to the more traditional document network). Users

have then evaluated the LIN to be similar to a tag cloud, while additionally giving a sense of the semantic contextual hierarchy [53].

Our data consists in a 12-year archive of daily Japanese public broadcast NHK News 7 [52], from 2001 until 2013. Each program is 30mn long with synchronized closed captions, about 6 months of 24/7 viewing in total. A program is composed of different *news segments*. As provided by Ide [33], we obtain the segments using a sliding window of topic distribution (Fig. 3.1, top). The extracted topics being very noisy, we further extract textual semantic information, for each segment, with a keyword extractor trained for news documents (including named entities) [26]. The Japanese keywords are translated to English with Bing Translator[1].

We use the face detection and tracking proposed in [52] (Fig. 3.1). Faces instances are detected in each frame [73], then regrouped with point tracking [63] creating *face-tracks*, sampled with k-faces [47], and represented using the average of its 128-dim OpenFace embedding vectors [3]. Face-tracks are clustered using GreedyRSC [40]. About 3,000 clusters are manually annotated, resulting in over 15,000 face-tracks of 139 public figures. We index each video segment with date-time of broadcast, keywords, and face-tracks. A query can be placed upon these criteria, returning a subset of news segments, including their associated semantic concepts, *i.e.* keywords and detected faces.

3.3 Multiplex abstraction

This section introduce our hierarchical clustering method. Subsection 3.3.1 introduce how we applied previous work, subsection 3.3.2 discuss a notion of similarity between concepts and groups of concepts, subsection 3.3.3 details our algorithm steps.

3.3.1 Layer interaction network and edge entanglement

Our input are the search results, which news segments associated with their bag of features (*i.e.* semantic concepts). "Graph of topics" [60] (or *LIN*[54]) are used to group results of a web search query. As pointed out, *"a graph of topics provides a contextualization of snippets"* [60], and enables us to measure group cohesion through multiplex *entanglement* [54].

Let S be the set of results returned by a query (see Fig. 3.1 (a)). Each segment in the results $s \in S$ is indexed by a set of concepts $t \in T$. We first consider a graph

[1]microsoft.com/translator

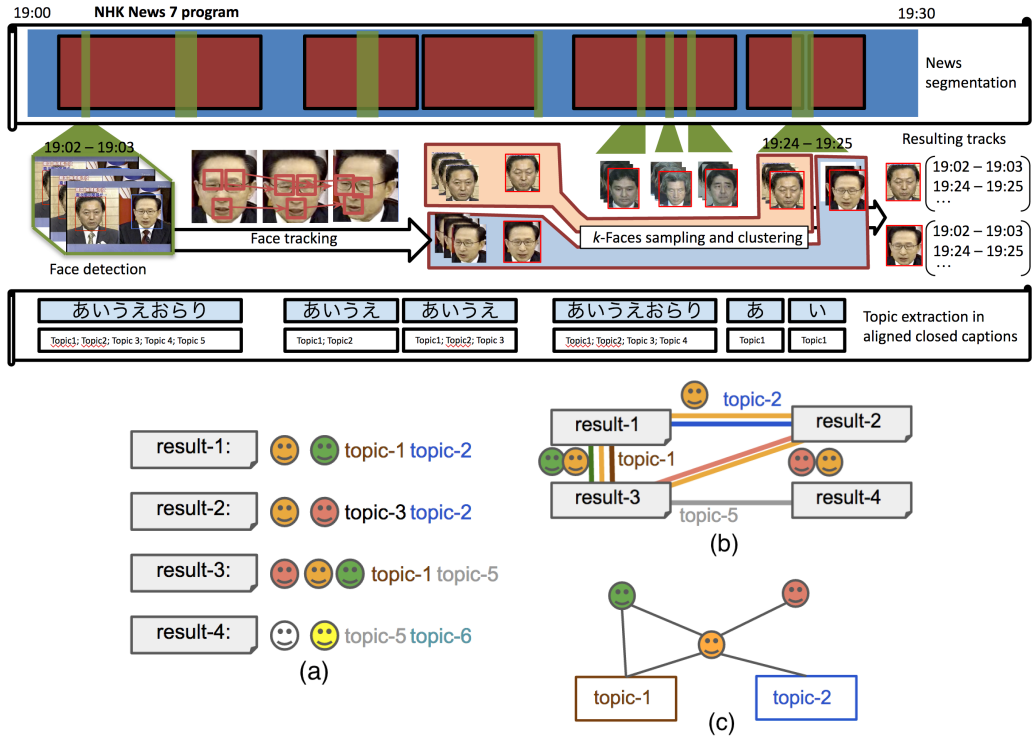


Fig. 3.1 (Top) Indexing each video segment: closed captions-based segmentation, face tracking, and keyword extraction. (Bottom) Abstraction of the search results; (a) indexed video segments; (b) the multiplex network of results; (c) the associated *LIN*.

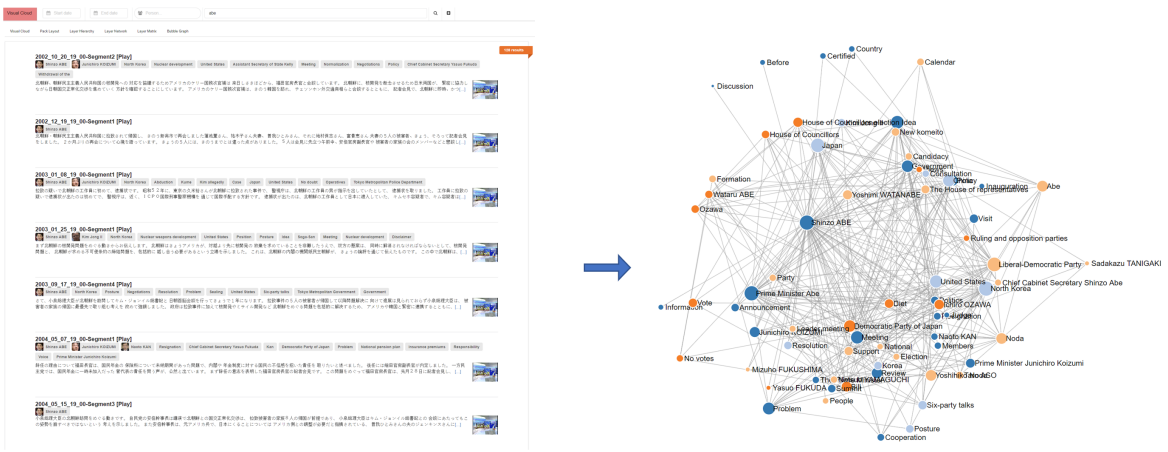
.

$G = (V, E)$ connecting segments to concepts. Two segments s, s' may share concepts $t, t', \ldots$ We build a graph where nodes correspond to segments $s, s' \in V$ and edges $e = (s, s') \in E$ are created if s and s' share at least one concept, e is *labeled* by the concepts $(t, t', \ldots)$ shared by s and s' (see Fig. 3.1(b)). This graph corresponds to a multiplex graph [54] $G' = (V, E')$ in which $E' = \bigcup_{t \in T} E_t$, *i.e.* each concept t forms a layer.

This formalism allows us to derive the *LIN*[54] $G_T\,(T, F)$, together with entanglement measures. The nodes of G_T are thus concepts $t \in T$, connected by an edge $f \in F$ if they overlap in G' (hence index at least two distinct segments). An edge f is weighted by $n_{tt'}$ the size of $|E_t \cap E_{t'}|$ which is the number of edges in G labeled by both t and t'. Following the same definition, $n_{tt} = |E_t|$ is the number of edges of G labeled by a concept t. Fig. 3.1(c) is a toy example of a LIN.

From there, Renoust *et al.* [54] compute an entanglement index γ_t for each concept: $\gamma_t.\lambda = \sum_{t' \in T} \frac{n_{tt'}}{n_t} \gamma'_t$. It measures how much concept t is entangled with other concepts in the network G'. In other words, it measures the share of concept t in mixing with

Fig. 3.2 Seraching results are built into interaction network



other concepts that brings results close one another. The more the concepts interact together, the more the group of results will be cohesive, and the more $\gamma_t \forall t \in T$ values will be similar. This is captured by the *entanglement homogeneity*[54] $\mathcal{H}$ which is then defined as the cosine similarity $\mathcal{H} = \frac{<1_T,\gamma>}{|1_T||\gamma|} \in [0,1]$.

3.3.2 Concept and group similarity

We now extend the work of Renoust *et al.* [54] by introducing a notion of similarity between concepts and groups of concepts: the *Co-occurrence Possibility* (CP). In G_T, concepts are connected when they co-occur through at least two distinct segments. The CP quantifies this association such as the higher the CP, the closer two concepts can be considered.

CP of two connected concepts t and t' in G_T is defined as follows:

$$CP\,(t,t') = \frac{n_{tt'}}{n_{t't'}} \cdot \frac{n_{tt'}}{n_{tt}} \in [0,1] \tag{3.1}$$

With n_{tt} and $n_{tt'}$ the number of (co)occurrences of the concept layers t,t' in the multiplex network. The higher the value of CP, the closer the two corresponding concepts will be. CP is maximized ($CP\,(t,t') = 1$) if and only if two concepts always appear together within results.

When t and t' are disconnected in G_T, we define a dissimilarity as follows:

$$CP\,(t,t') = -\frac{n_{tt} + n_{t't'}}{|E|} \in [-1,0] \tag{3.2}$$

With $|E|$ the number of connected pairs of nodes in the multiplex network. The CP value is then negative, with minimal value $CP(t,t') = -1$ when the two concepts t and t' cover all the links between segments but never co-occur together, hence representing two separated topics.

Following the same idea, the CP between a concept t and a group of concepts S is formulated as:

$$\sum_{t \in S} CP(t,S) = \sum_{t' \in S} CP(t,t') \ , \ t \neq t' \tag{3.3}$$

Finally, the CP between two groups of concepts S and S' ($S \neq S'$) of concepts is then defined as:

$$CP(S,S') = \sum_{t \in S} CP(t,S') \tag{3.4}$$

We may verify that $CP(S,S') = CP(S',S)$. This formulation of similarity could be seen as analogous to forces of a force directed layout algorithm[49]. Similar concepts should tend to "attract" each other, while dissimilar concepts will "repulse" one another, placing them far apart: high negative avoid unconnected concepts to be assigned to the same group.

3.3.3 Layer hierarchization

In Detangler [53], the *LIN* captures the semantic context with *"a sense of hierarchy"*. However, the network drawn "as is" tends to quickly become a furball as the number of concepts increases. Hierarchical relationships can provide a meaningful navigational mechanism by organizing information into a small number of hierarchical clusters [80]. To extract this hierarchy, our strategy is to identify concept and segment subsets with optimal cohesion (entanglement). Since it is an NP-hard combinatorial optimization problem, Renoust *et al.* [54] did not offer any solution. Keeping in mind that we have a subgraph of segments corresponding to each (group of) concept(s), we propose a heuristic solution to maximize cohesion in groups of segments.

This algorithm aims at building trees of concepts from the *LIN* G_T. Each concept starts labeled with its own group l_t and we aggregate concepts from the topology of G_T such as two concepts will be connected if they are linked by an edge in G_T. Orientation of links are decided from $max(\gamma_t, \gamma_{t'})$ (rooting on the most entangled concepts). The algorithm runs in three steps (Fig. 3.3). The first step aims at initializing parent-

children links. The second step associates all children nodes to their closest concepts (or group of concepts). The last step iteratively assembles groups of concepts.

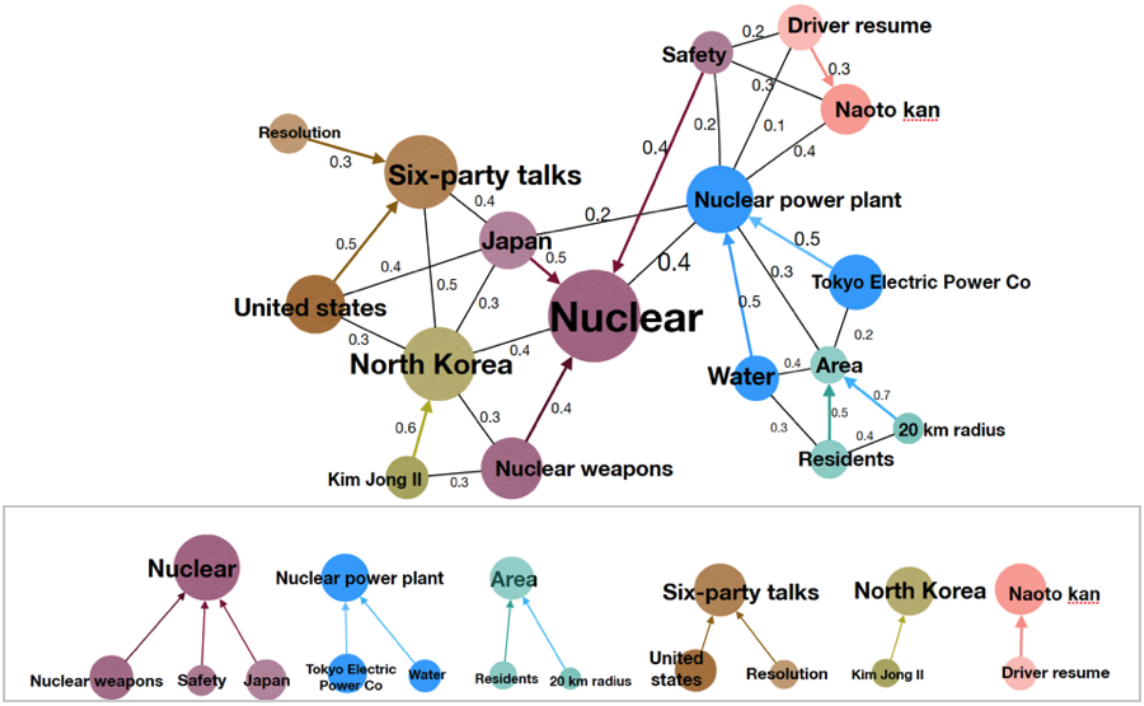


Fig. 3.3 Step 1: constructing the first trees by pair association.

Step 1: Generate "parent-children" pairs for all concepts. In this step (Fig. 3.3(a)), each concept t_1 , that is not a parent already, will be associated to a concept t_2 in its neighborhood $t_2 \in \mathcal{N}_{G_T}(t_1)$ such as $CP(t_1, t_2)$ is maximal. All the pairs form now C groups, each node will then be labeled with its parent's group l_p.

Algorithm 1 Generate "parent-children" pairs for all concepts

procedure INITIALIZATION
  for t in T **do**
    assign to t a unique label l_t
procedure STEP 1
  for t in T **do**
    if t is not *parent* **then**
      $t_2 \leftarrow n \quad \backslash \quad \max_{n \in \mathcal{N}(t)}(CP(t, n))$
      if t_2 is not parent **then**
        $x \leftarrow parent \quad \backslash \quad \max_{x \in \{t, t_2\}}(\gamma_x)$
      $l_y \leftarrow l_x \quad \backslash \quad x$ is parent of y, $x, y \in \{t, t_2\}$

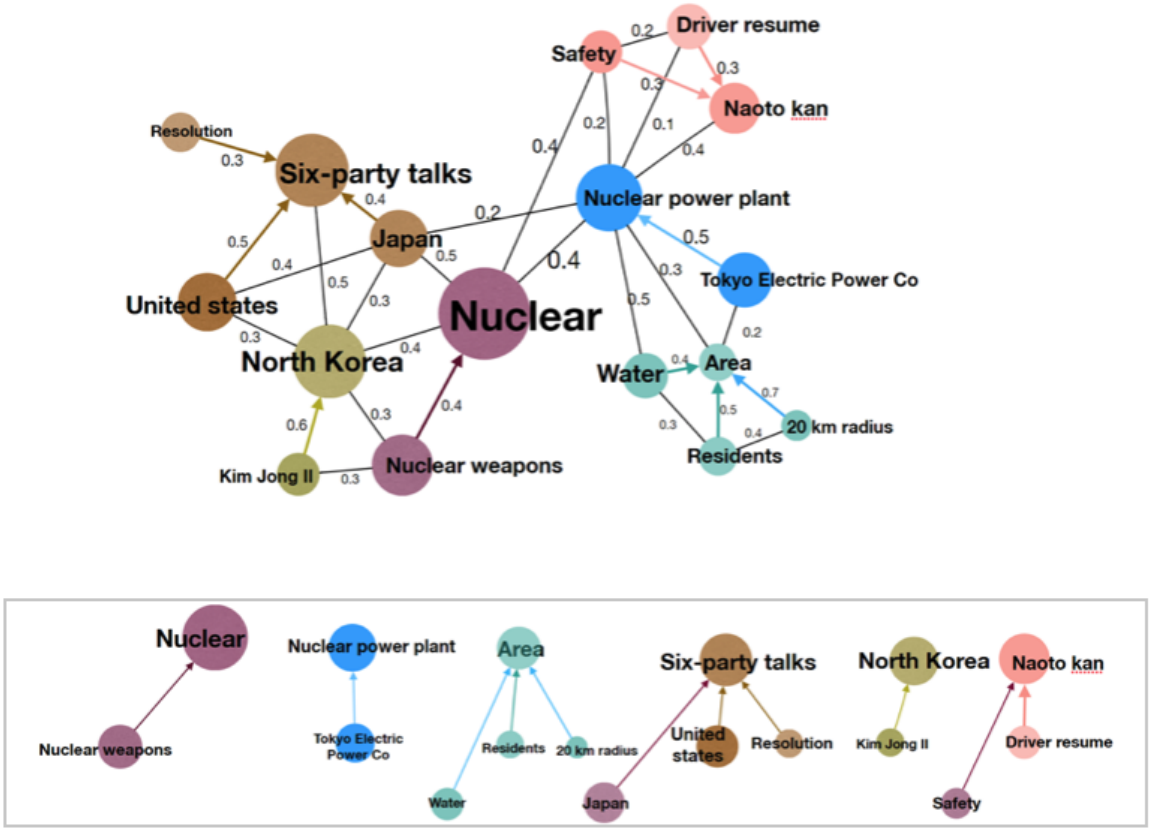


Fig. 3.4 Step 2: moving nodes to optimize group assignation (highlighted in red).

Step 2: Link children to their closest concept or group of concepts. We need now to associate concepts to the group in which they can maximize entanglement. We now compare the group $c \in C$ in which a concept t maximize $CP(t, c)$ to the neighboring child concepts t_2 that also maximizes $CP(t, t_2)$. A pair of nodes may maximize together entanglement in comparison of those of a group, so we create in this case a new parent node. In other words, this avoids too coarse aggregations often induced by very occurring concepts, such as queried criteria. We continue updating associations until no change occur anymore. Results are illustrated by in Fig. 3.3(b).

Step 3: Regroup all groups in a forest. After the second step, all concepts are now part of a parent-children association that took into account group CP (with positive and negative weights). We now compute the hierarchy between group, by simply comparing CP of groups together. Similarly, we will associate neighboring groups depending on their CP, starting with the pair of neighboring group displaying the highest CP in the graph (Fig. 3.3(c)) (and if there is a connection between them). We may obtain multiple hierarchical trees and the most obvious case is when the *LIN*

Algorithm 2 Link children to their closest concept or group

procedure STEP 2
  $C = \{C_1, C_2, \cdots, C_k\}$ (groups formed from step 1)
  repeat
    $changed \leftarrow False$
    for t in nodes $\setminus$ t is not *parent* **do**
      $c \setminus \max_{c \in C}(CP(a, c))$
      $t_2 \setminus \max_{t_2 \in \mathcal{N}(t)}(CP(t, t_2))$, t_2 is not *parent*
      if $CP(t, c) > CP(t, t_2)$ **then**
        if $l_t \neq l_c$ **then**
          $l_t \leftarrow l_c$
          $changed \leftarrow True$
      else
        $x \leftarrow$*parent*, $x \in \{a, b\} \setminus \max_{a,b}(\lambda_a, \lambda_b)$
        l'_x, new label
        $l_y \leftarrow l_x \leftarrow l'_x$
        $changed \leftarrow True$
  until $\neg changed$

Algorithm 3 Regroup all groups in a forest

procedure STEP 3
  repeat
    $c_x, c_y \setminus \max_{c_i, c_j \in C}(CP(c_i, c_j))$
    $u \leftarrow max_{k \in \{x,y\}}(\gamma_{Parent(c_k)})$
    $v \leftarrow min_{k \in \{x,y\}}(\gamma_{Parent(c_k)})$
    $l_{c_v} \leftarrow l_{c_u}$
    $Parent(c_v) \leftarrow Parent(c_u)$
  until $|C| = 1$ or $max_{C_i, C_j \in C}(CP(C_i, C_j)) <= 0$

is not always only one connected component. We stop when there are only negative distances (*i.e.* $\forall i, j\ CP(c_i, c_j) \leq 0$) or only q groups ($q = 1$ or user specified).

The overall structure of our cluster algorithm is very similar to the method introduced by Blondel et al. [13]. Their work proposes a simple method to extract the community structure of large networks. The complexity of our grouping algorithm is in $O(E)$ where E is number of edges.

This bottom up hierarchical cluster method has three features:

Dendrogram. In the first and the second step, the node with a relatively large weight is always used as the parent. In the third step, when two different groups are merged, the parent with the larger weight is also selected as the root of the new group. Each root(parent) node is the node with the highest weight in the group. The existence

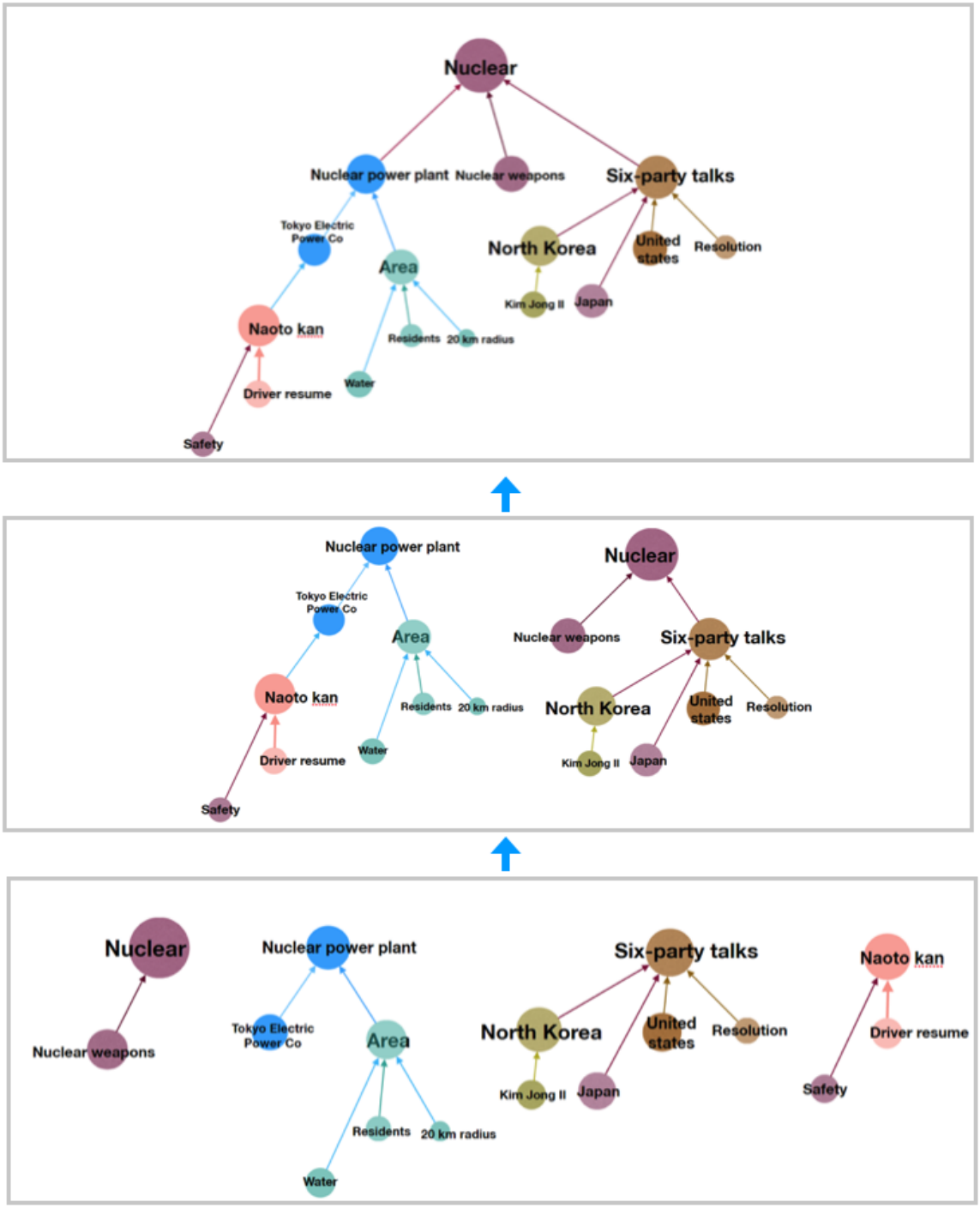
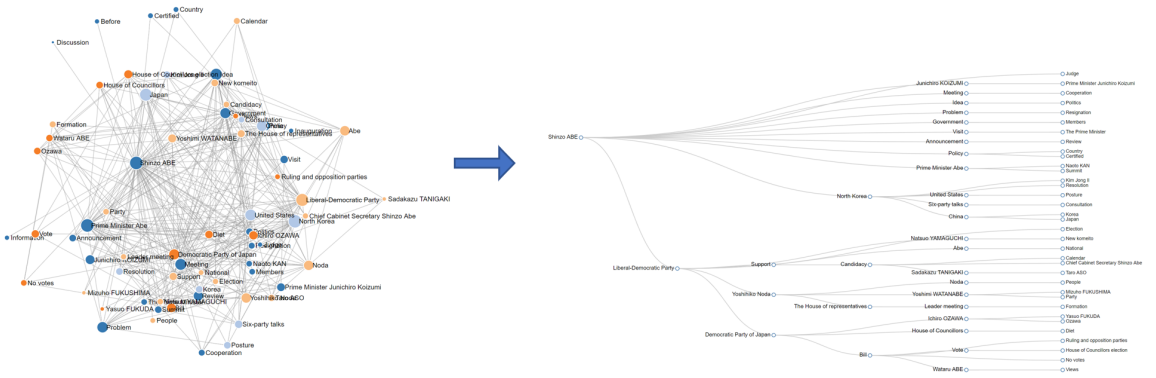


Fig. 3.5 Step 3:aggregating groups to construct the final hierarchy.

of the parent(root) node can clearly show the hierarchical relationship between the concepts, which allows the user to understand the search results more systematically. As 3.6 shows, the hierarchical cluster results from query "ABE".

Fig. 3.6 Concepts are clustered into dendrogram.



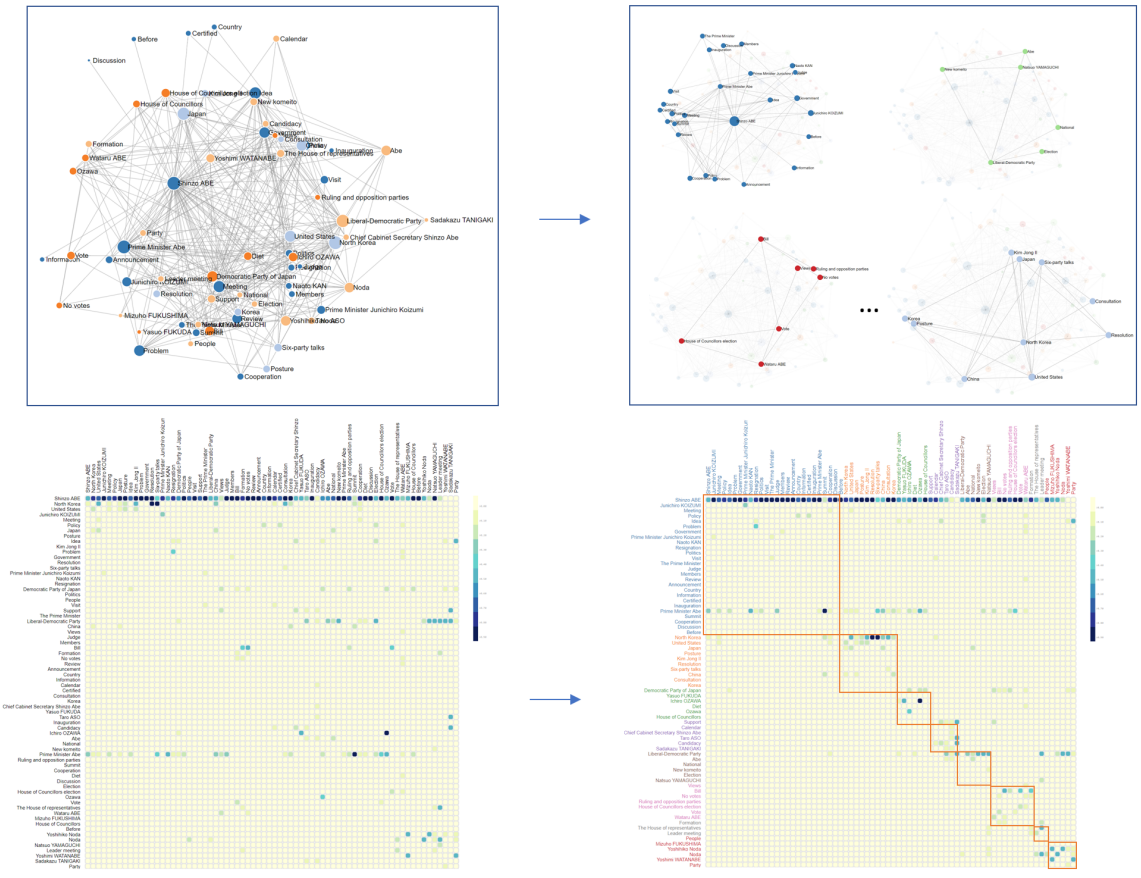
Get the best number of groups. Repeat the third step until the CP value between all the different groups is less than zero, and the number of groups is the optimal number. As 3.7 shows the adjcency matrix is supported in our framework, it can also be used to test the quality of group results.

Get the required groups number. When get a lot of parent-child like groups through the first and second steps. In the third step, the two groups that are most closely connected are merged. In the process of merging, if the number of groups set by the user is q, when the number of groups reaches q, the merging process ends, and the number of groups that the user wants is obtained. If the number q set by the user is less than the optimal number of groups obtained by the algorithm, in the third step, if the CP value between any two groups is less than zero, but the number of groups is still greater than the number required by the user, the merging process will continue. Perform the merge of the two groups with the largest CP value, even if the CP value between the two groups is less than zero. Through this forced merger, the amount required by the user is finally obtained. As 3.8 shows, different colors represent different communities, and users can divide the same graph into any number of required groups. Of course, the number of groups is less than or equal to the number of nodes.

3.4 Visualization of heterogeneous clouds

Many attempts to display overview or grouping information have focused on automatically extracting the most common general themes that occur within the collection. In document clustering, similarity is typically computed using associations and commonalities among features, where features are usually words and phrases [8]. The greatest

Fig. 3.7 The adjcency matrix display the grouping results



advantage of clustering is that it is fully automatable and can be applied to any text collection without manual [9].

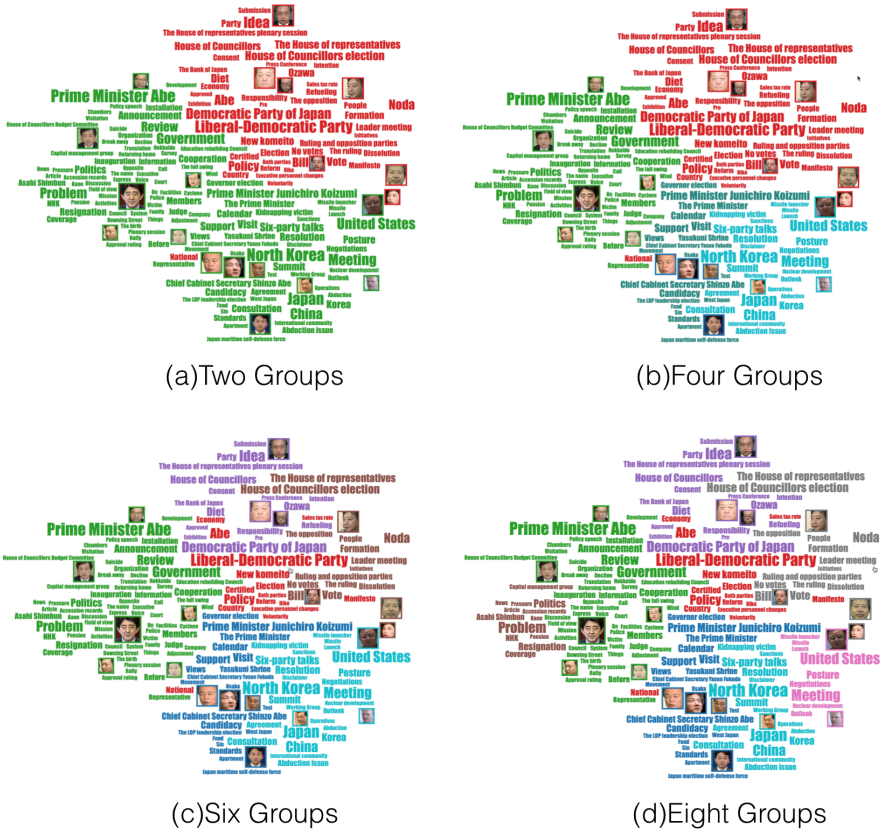
Queries on our news search engine return a set of video segments, but since the activities of public figures can span over long periods of time and diverse topics, we need to provide contextualization on-the-fly.

3.4.1 Visual cloud generation

To best support users, the visual cloud should be compact, aesthetic and expressly map the thematic grouping resulting from the hierarchical structure extracted. Inspired by usual tag cloud, it should also integrate images seamlessly. We start by embedding the hierarchical relationships.

Pack Layout initialization: The Pack Layout algorithm [74] uses enclosed diagrams to represent containment (nesting) as the hierarchy (similarly to treemap

Fig. 3.8 User could group the tags into specific numbers, The different colored tags in each figure represent different groups.



algorithms). The size of each leaf node reveals a quantitative dimension associated to data points and the enclosing circles show the approximate cumulative size of each subset. Circle packing does not use space efficiently, however it can indicate relative positions of nodes following their hierarchical relationship. Each concept is assigned to the center of the enclosing circle it is represented by (Fig. 3.9(b,c)). Pack Layout creates a leaf-node per node in the tree, instead, we assign parent nodes to the center position of the higher order circles. This results in a more even distribution of position in the plan, giving a relative position for each concept guided by the hierarchy (Fig. 3.9, left).

Visual cloud layout: Wordle algorithm [72] is arguably the fastest tag cloud algorithm. Words initial position can not be strictly specified, and size depends on words relative frequency. Words are introduced one by one to some random position close to the center of the canvas and iteratively placed in the order of frequency. A word is then displaced if it intersects with any previously positioned words. This displacement

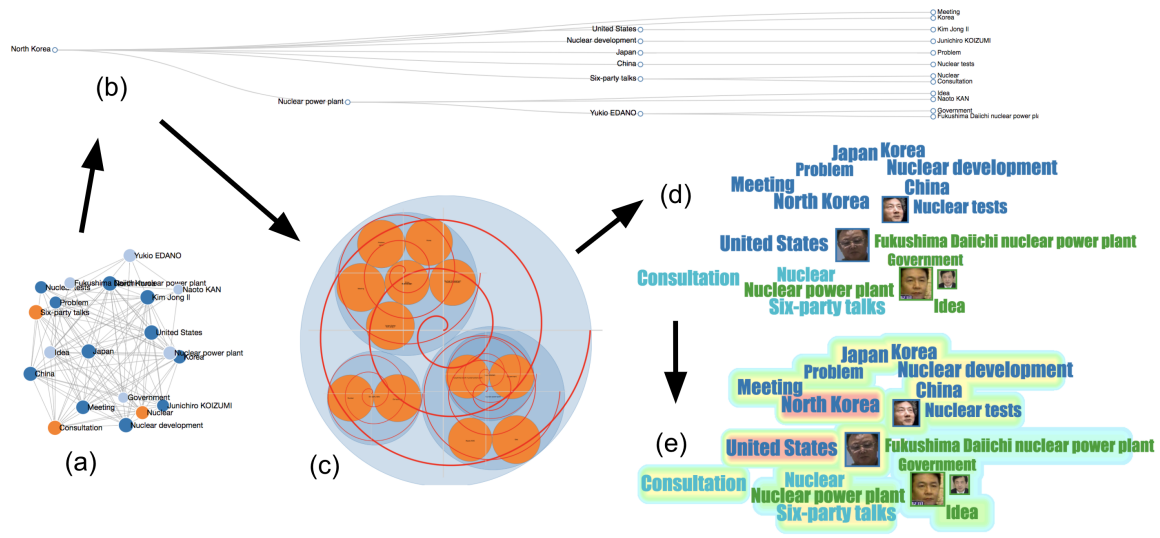


Fig. 3.9 Visual cloud layout: (a) *LIN* (b) extracted hierarchy (c) Pack Layout embedding (with spirals) (d) visual cloud (e) with heatmap.

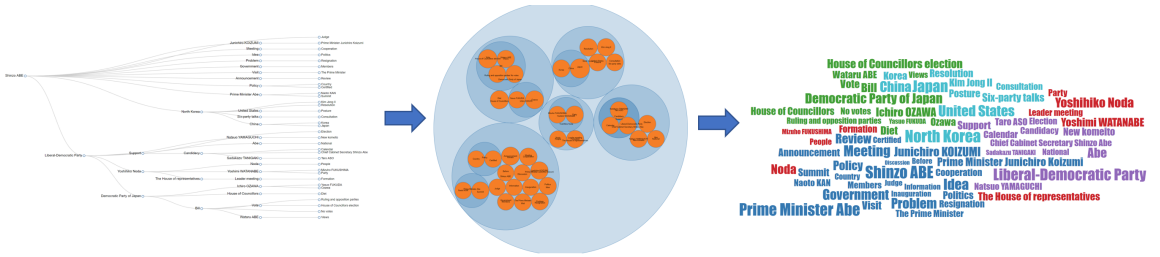


Fig. 3.10 Dendrogram is used to build pack layout, the location of the tag in the tag cloud depends on the location of the tags in the pack layout.

is made following an increasing Archimedean spiral until no more intersection is found. We use this to our advantage by constraining the spirals with the Pack Layout's circles (center and separation distance), see Fig. 3.10. We extended the algorithm to take into account any rectangle shape. As a result, concepts are placed in the proximity of their previously calculated position, relatively reflecting their hierarchical structure (Fig. 3.11).

Regions of Interest: To highlight regions of interest, we offer multiple visual encoding. Textual concepts and image borders are colored upon the group to which they belong.

The number of groups can be interactively set, since it corresponds to a different cut of the hierarchy, it simply updates colors, while keeping the layout stable. The size of a concept encodes its frequency on the edges of the multiplex network. Because frequency is only one aspect of the significance of a concept in its group, we introduce a

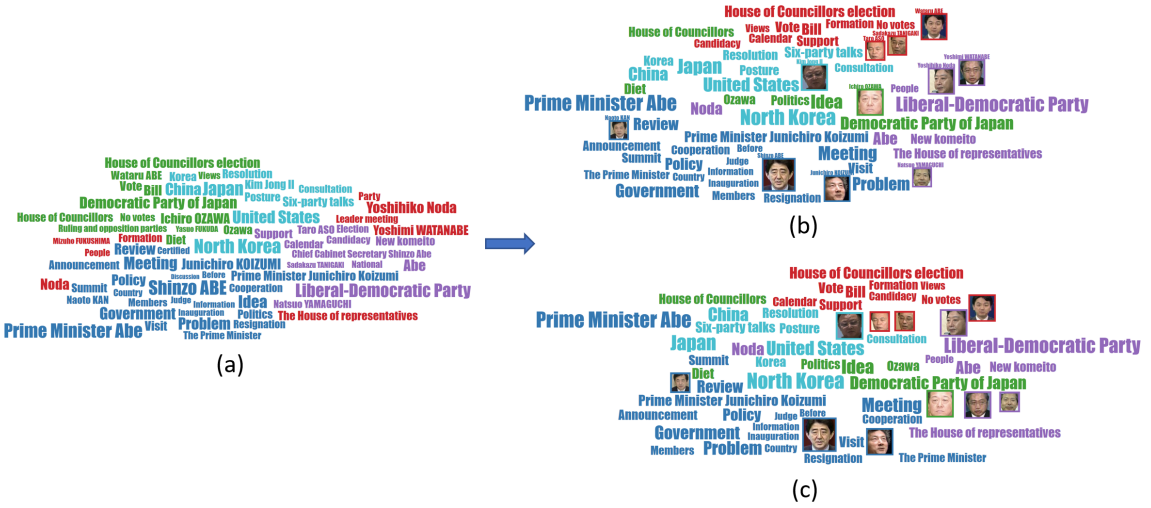
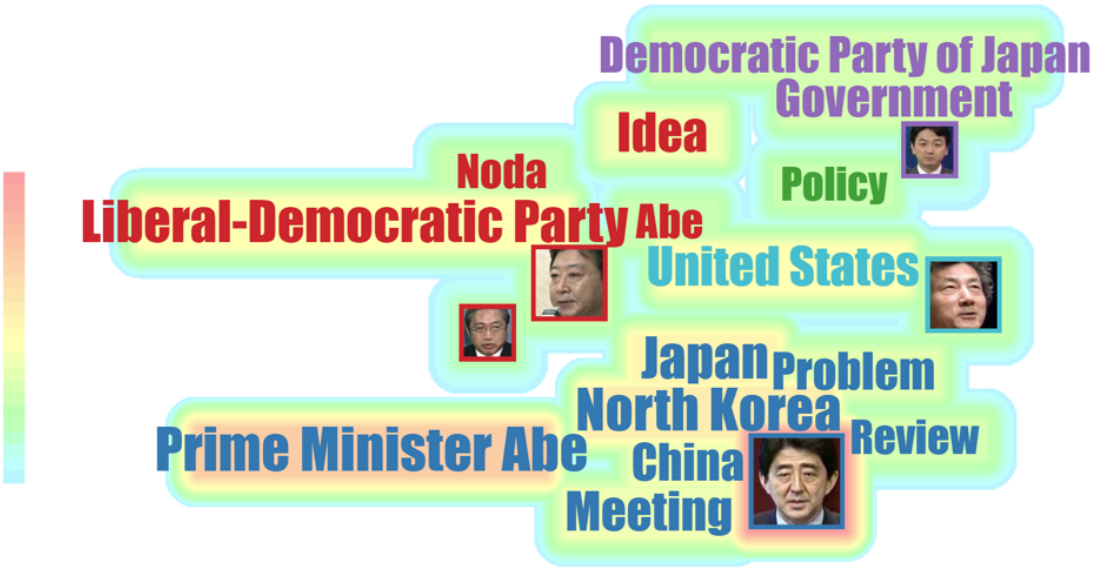


Fig. 3.11 Since the position occupied by the tags is a rectangle and the character image is also a rectangle, we use the character image instead of the tags in the process of calculating the layout, so that the tags and the image can appear at the same time, and the hierarchical placement is kept. (a) displays the normal tagcloud, we use the person's name instand of their profile photo, (b) displays the profile photo and the name at the same time, (c) displays only profile photo

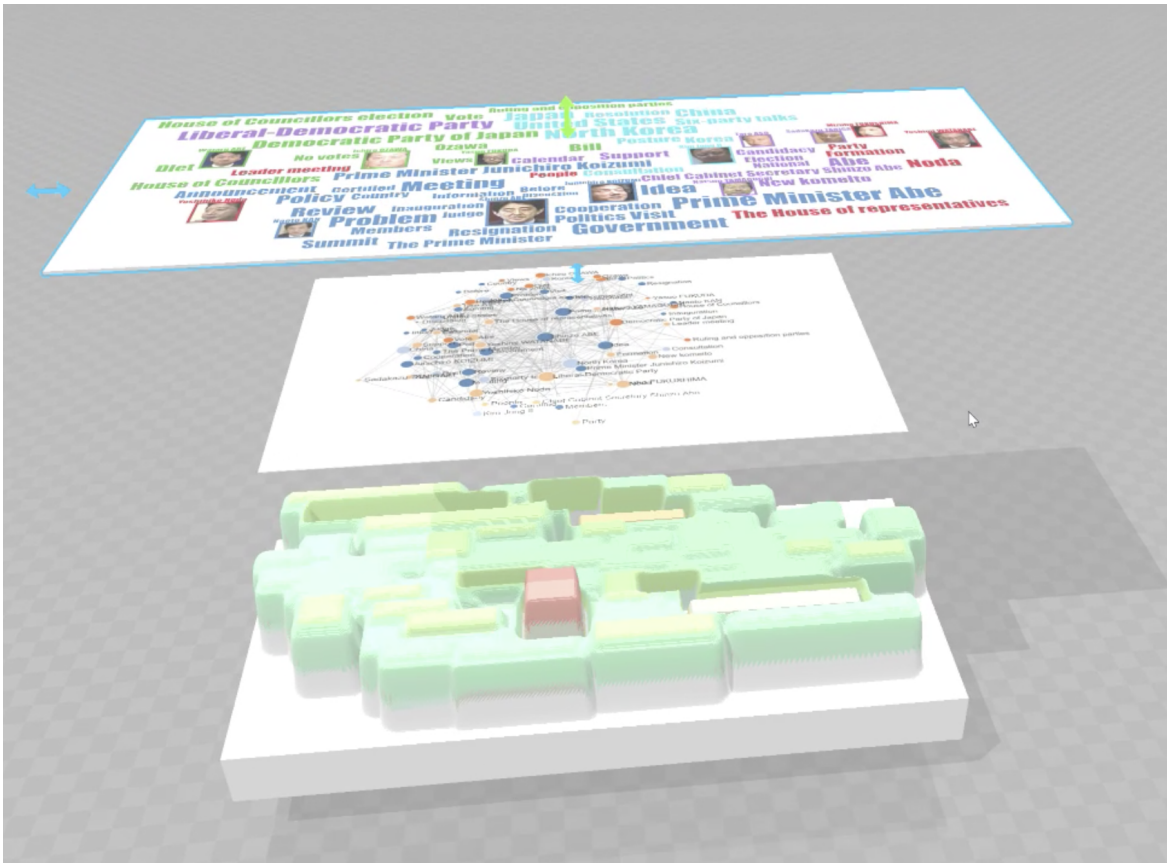
Fig. 3.12 A canvas made heat-map is used to show the importance of concepts.



new highlighting that displays how concepts mix with others: an optional background heatmap displays the entanglement index of concepts (Fig. 3.12) and contrasts enough

with the categorical scale. Higher entanglement indices assigned to warm color attract user attention. A *max* blending function as the heat value diffuses from the outer box of concepts maintains text readability.

Fig. 3.13 There are two layers behind the tags cloud: a network and a heat-map.



Users can place advanced queries on the three criteria of time-frame, face and keywords (Fig. 3.14 (a)). Above the list of results and the visual cloud (Fig. 3.14 (d, c)), the system shows a brushable time bar chart that positions the query results in time (the timeline background is tuned to our usage scenario to show periods of interest in color). Similar to traditional search engines, a list of results is presented (Fig. 3.14 (d)). It is ordered by time, titled by date/time/segment. A snippet composed of the first lines of captions can be expanded. Clicking on a result launches a video player at the segment position.

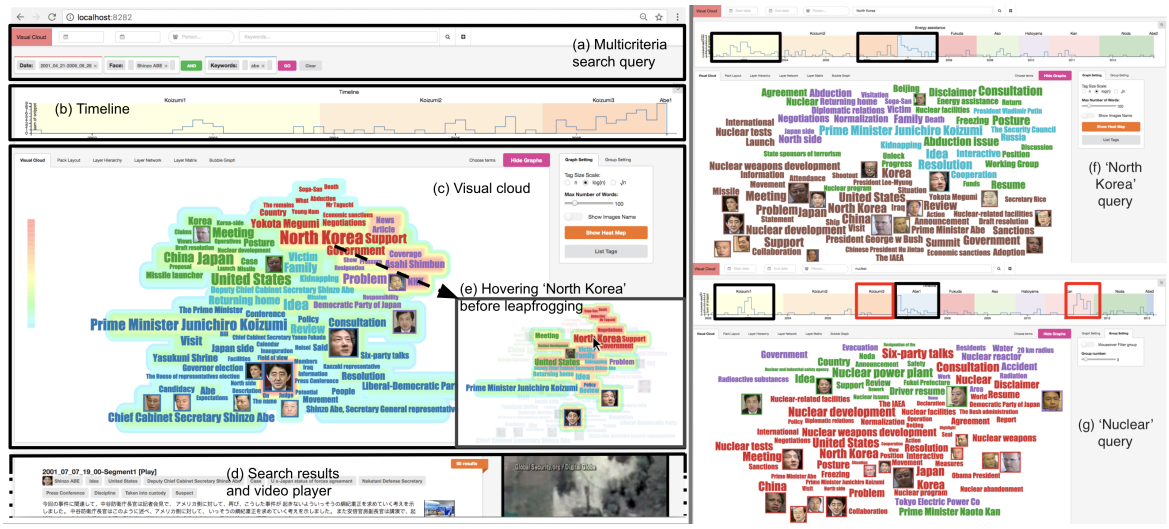
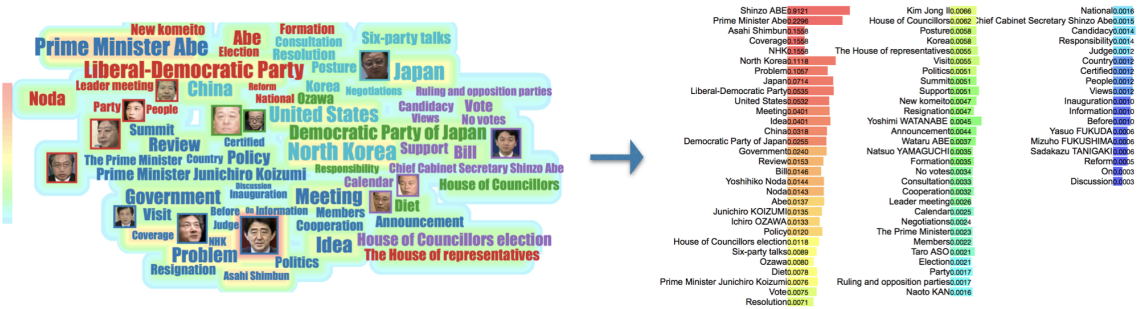


Fig. 3.14 Left. Interface overview, search query over Abe during Koizumi terms (a) - very active during the end of Koizumi3 (b), mention of 'North Korea' stands out (c). Hovering over 'North Korea' shows that it strongly relates to 'abduction' (e). Right. Comparison of the queries on *North Korea* (f), and *Nuclear* (g): two periods coincide (in black) Koizumi1 and Abe1 but we can notice two major differences (in red). Koizumi3 period did not associate much *North Korea* and *Nuclear*, and Kan period associate *Nuclear* with *Accident*. Demonstration video available at https://youtu.be/VfGwa6T94t8.

Fig. 3.15 An optional list of concepts view could be displayed. Concepts are ordered by its entanglement index, the bar after each concept represent their degree in the graph, their color is the color of their heat in heat-map which is based on their entanglement index.



3.4.2 Interactions

Filter the concepts Filtering results is achieved by clicking a concept or brushing the timeline. As shown in the figure, user can select a period of time through brushing

timeline. The curve of the number of snippets on the timeline help user to make chocie. After selecting a certain period of time, the interaction network will be re-computed, showing the tag clouds in that time period, and the size of the tags will be dynamically adjusted, so that users can understand the changes of concepts more deeply. however, *leapfrog* interaction [53] by double-clicking a concept or the timeline brush will result in a new query corresponding to this filter, recomputing a new visual cloud for a finer grain analysis. This corresponds to *Search* tasks often formulated by Shneiderman's mantra *"Overview first, zoom and filter, then details-on-demand"* [64].

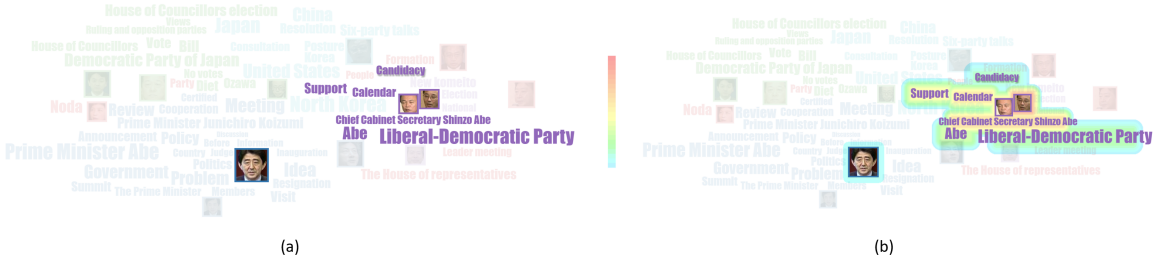
Fig. 3.16 Select different time interval from timeline, the concepts only appear in that period will be displayed, the concepts' size would be changed.

Mouse over interaction Extracting a hierarchy from the *LIN* implies that we loose its topology but the neighborhood of a concept is key to exploration. It is restored through hovering interaction: all concepts are dimmed except for those neighboring the concept in the *LIN* with timeline highlighting (Fig. 3.14(c,e)). When the heatmap is active, new heat values are computed on-the-fly which map the *local* entanglement indices of concepts in the multiplex subgraph induced by the concept hovered (similar to [53], Fig. 3.14(e)).

3.4.3 Query system and implementation

The three important elements for labeled TV news videos are: "when", "who", "what". In our interface, user could search TV news by time (when), person's name (who) and keywords (what). Also the boolean query is supported for advanced search. User could use multiple query operators and quotation marks for boolean search. As figure 3.18 shows the example of boolean query $a \vee (b \wedge c)$.

Fig. 3.17 Mouseover interaction. (a) Mouse move over the tag 'dandidacy', only connected tags displayed, the other tags become pale. (b)When heatmap is displayed, the new heatmap will be redrawn based on the entanglement index of the subgraph of the mouseover tag.



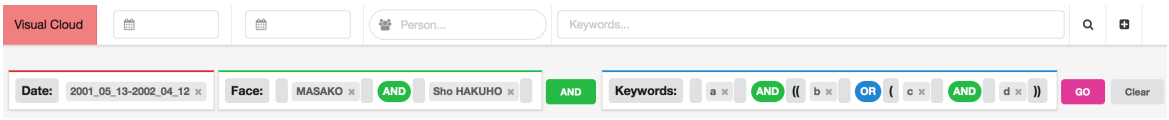


Fig. 3.18 User could search by inputing time period and person's name and keywords, figure shows a boolean query example.

User could select time period by simply drag the brush of time-line, also, user could modify the boolean query by simply clicking the options. Those feature make easier for user to select the desired time clip and results.

Implementation and complexity: The system is implemented in HTML5 with the popular semanticUI [61] and D^3 [23] libraries. Database indexing and access is implemented in python. The hierarchy algorithm complexity is bounded by the computation of entanglement which requires an Eigen decomposition of a matrix of dimension the number of concepts.The construction of the LIN and $n_{t,t'}$ are made while constructing the multiplex network of results. Both depend on the number of results $|V|$ and number of concepts $|T|$, $O(\frac{1}{2}(|V| * (|V| - 1) + |T| * (|T| - 1))$. The computation of the hierarchy is a greedy optimization similar to Louvain [13] estimated in $O(|T|log|T|)$. No complexity is discussed for the Pack-Layout algorithm [74] but runs in milliseconds for a thousand circles. The number of circles depends on $|T|$, which is at best a few hundreds. The word cloud generation is based on Davies' heavily optimized implementation (see www.jasondavies.com/wordcloud/about/, which is bounded by computation of bounding boxes and collisions (not impacted by our modifications). Including the Pack-Layout initialization, the word cloud generation can be considered instantaneous. The heat-map generation is done in one pass through each concept,

with a static Canvas implementation faster and more memory efficient than DOM population of SVG elements.

3.5 Evaluation and use cases

We have made use studies to evaluate our cluster algorithm and the visual cloud. We invited students to use our framework, and made a comparision to other cluster methods and tags clouds.

3.5.1 Usage scenario, Abe and the North Korea:

In a previous study, the authors investigated appearances of Japanese Prime Ministers on NHK[52]. One interesting conclusion was the growth in screen appearances of Abe during Prime Minister Koizumi's ruling, before becoming himself Prime Minister in the following elections. Our system allows to refine this study by placing a complex query to search all news segments during Koizumi mentioning Abe by name or face (Fig.3.14). A demonstration video is available at https://youtu.be/VfGwa6T94t8.

The timeline (which is augmented with Prime Minister's rulings on the background) confirms the growing mention of Abe. This is no surprise knowing that Abe was chief cabinet secretary during Koizumi's third term. The visual cloud proposes 5 groups: about elections, about Yasukuni Shrine, about the newspaper Asahi Shimbun, about Japan/Korea/China, and about North Korea. But turning on the heatmap, the most prominent word becomes North Korea by far. Indeed, Abe was chief negotiator on issues related to abductions of Japanese citizens by North Korea, managing to free 5 of them. Leapfrogging on the keyword "North Korea" (Fig.3.14(e)) makes a new search of Abe associated with North Korea during Koizumi terms. Browsing the timeline among the three terms highlights different subtopics at each terms. We may mention a meeting during Koizumi's first term, associated with the three faces of Koizumi, Kim Jong Il, and Abe; mentions of sanctions, missile launch and draft resolutions during Koizumi's third term.

Now a new search on the keyword *North Korea* highlights that it is most active during first Koizumi ruling and especially during first Abe ruling (Fig.3.14, left). North Korea related abductions was excessively reported on the media and Abe's administration has put pressure on NHK to "pay attention" [43]. A last search on the keyword *nuclear* gives 3 spikes in the timeline (Fig.3.14, right). Two of the spikes coincide with *North Korea* previously described (Fig. 3.14). One big difference comes

with Koizumi's third term, when no mention of nuclear issue is made. The last in 2011 after the Great East Japan Earthquake about the nuclear powerplant accident.

Validation of the hierarchical algorithm: We offer a heuristic that optimizes entanglement homogeneity $\mathcal{H}$ in networks formed by association of results and concepts that captures cohesion of a group of documents[54]. To the best of our knowledge no other work attempt create a hierarchy of concepts to maximize cohesion between documents related to concepts on one hand, while preserving the relationships in the *LIN* on the other hand. However, we can still compare $\mathcal{H}$ with that of groups of documents corresponding from concept clusters induced by a Louvain segmentation [13] on the *LIN*, and with random segmentation to serve us as a baseline. The random segmentation only agglomerates nodes randomly from the network's topology until reaching a given number of clusters. Based on 8 queries of 50/100/150/200 concepts, we randomized 10 generations of k clusters (k reused from the Louvain segmentation [13] to equally compare between the three types of segmentation). We then average $\mathcal{H}$ for each group, across queries and segmentation. Results in Table 3.1 confirms that our segmentation results in more cohesive subgroups.

Table 3.1 Comparing $\mathcal{H}$ among groups of results of varying sizes

Group Size	*Random*	*Louvain*	*Multiplex*
All sizes	0.43874	0.53359	**0.59600**
50 concepts	0.58213	0.66137	**0.68647**
100 concepts	0.38544	0.46707	**0.55022**
150 concepts	0.38665	0.52803	**0.60585**
200 concepts	0.40075	0.47792	**0.54149**

3.5.2 Evaluation of the hierarchical word cloud

To evaluate the output of visual cloud, we conducted a user study. We compared the output of our algorithm with three other word cloud algorithms that focus on the preservation of semantic relationships. The first, *"Inflate-and-Push"* (IP), is a semantic preserving word cloud based on multi-dimensional scaling [10]. The second is the *"context preserving word cloud"* (CP) [22]. The last, *"Star-forest"* (SF) is the closest to our spirit [10, 12]. We used for the three of them a 4:3 image ratio with cosine similarity for relationship between words and term frequency for word ranking. The implementation used is proposed by the university of Arizona[2]. Incidentally, Louvain segmentation [13] is also the way the groups of words are chosen in this implementation.

[2]See `https://github.com/spupyrev/swcv`

Table 3.2 Comparison of cluster results

word cloud	visual cloud	IP	CP	SF
all sizes				
grouping	2.0625	2.5938	2.5938	2.7813
location	1.7188	2.9375	2.1875	3.2813
preference	2.0625	2.6875	2.2188	3.1563
50 words				
grouping	1.75	2.5	3.125	2.75
location	1.25	3.5	2.125	3.125
preference	2.125	2.5	2.375	3
100 words				
grouping	2.25	2.125	2.5	3.125
location	2.25	2.375	2.125	3.5
preference	2	2	2.375	3
150 words				
grouping	2	2.875	2.25	2.75
location	1.875	3	2.25	3.125
preference	2.125	3.125	1.875	3
200 words				
grouping	2.25	2.875	2.5	2.5
location	1.5	2.875	2.25	3.725
preference	2	3	2.25	3.25

In total we evaluated 32 queries among 8 different users, each query with 4 different representations. We generated 8 different search queries that may contain ambiguous results: *swift, apple, jaguar, serendipity, ring, network, orange* and *uncertainty* and gathered about 250 snippets for each query. The evaluation was conducted with the help of college students, each student had 4 queries (each of varying size from 50 to 200 words) with the four different word cloud representation to evaluate on a query. The presentation of the word clouds have been shuffled to avoid ordering effect, and the overall interview took about one hour each (including informal interview after).

We ask them to rank the 4 word clouds by three criteria:

- meaningfulness of the color grouping (do words in a group really belong to a group?);
- meaningfulness of the positioning (do close words well relate to each other?);
- preferred word-cloud (including aesthetics).

We average the ranking among all queries on each criteria and results are shown in Table 3.2. The meaningfulness of our grouping and the meaningfulness of our words

positioning have received the best results. Although overall users were preferring our word cloud over others, two users clearly did not so. The grouping appeared to be quite disturbing and too aggressive for them, as they would like to see appearing more ontological grouping rather than topical. A user also noticed too many non-meaningful words such as “likes” or “related”. This suggests that we need to improve our pre-processing as part of our future work.

We also broke the results according to group size in Table 3.2 (8 experiments per group of 50). It is interesting to note that our algorithm performs particularly well for location meaningfulness of the words, especially for the large groups. This is a comment confirmed while we interviewed users, as our grouping technique was helpful in breaking down large numbers of words. Note that we were slightly outperformed by CP for location and IP for grouping meaningfulness for 100-word queries, and by CP as well for 150 words as users’ preferred word cloud. CP tends to better separate its clusters for the queries at 100 words than it did with other queries (often mixing words of different clusters). As for the users preference, CP produced the less compact layout of words by allowing blank spaces, which especially pleased some of our users.

3.5.3 Usefulness of the representation

We conducted an informal study to get feedback on the usefulness of our system. After each experiment described previously, we interviewed preference on with/without heat map on word tags in relation to size. Users almost always preferred the our word cloud generated with heat-map.

Interviewing our users on their experience, they tend to emphasize on the usefulness of the hierarchical layout, and their usage meant often turning on and off the heat-map representation. The heat-map appears as an optional feature to them. It was also noted that it sometimes disturbs the reading of text. However, users also reported that they were able to understand the thematic from the word cloud faster by using the heat-map feature. When a large number of words was presented, they reported to also better spot important words.

We further let users do 10 search queries each and collected feedback of their user experience in comparison with a regular search engine results:

- Most users have used our topic map to help search for results when not the result they were hoping for was not clearly found within the first two pages of search results.

- When users are not familiar with the area they are searching in, they prefer using our interface. For example some users preferred using our interface to search about celebrities, places, news.
- All users are interested in the interactive re-ranking of results and would like it implemented in search engines.
- Word cloud with highlight is obviously more readable than their own experience of common tag cloud. However, with complex query like with statement, they would prefer sentences rather than terms to understand the search result.

The feedback is very important. It highlights that users have many different behaviors and intentions when searching the web. This is well captured by Brehmer *et al.*'s typology [15] (in their Fig. 1), in the searching task: search location and search target can be known or unknown. This corresponds to four different behaviors: *lookup, locate, browse* and *explore*. Although we designed this system for *explor*ation tasks, we have not thought of differentiating those cases before conducting our experiment, it clearly impacts the usefulness of ours system.

A one-cent guess from this informal feedback we collected is that search engines are often used for *lookup* tasks (with location and target both known). In this case the word cloud representation is certainly less useful. While fully making sense for *explor*ation tasks, usefulness of our representation then rises when *brows*ing, but less for *locat*ing tasks. Indeed, in *locat*ing tasks, users expect the target to fall under the first few search results. However, in real cases, there is a delicate balance between both location and browsing cases: users do not always have full certainty of the target to find, or its location.

3.6 Conclusion and perspectives

Although tag clouds do not offer much room for visual encoding except for layout, size, and color, users have positively welcomed the heatmap. It helps to correct the information overload when too many concepts are displayed, by bringing initial focus on the highlighted concepts. However, it reduces the perception of group differences. Future work will explore the design space offered by joining heatmaps and visual clouds, especially for interaction (used here to reintroduce the *LIN* topology). We also improve traditional tag clouds with thumbnails as a supplementary information, bringing new information in form of visual cues. We only use faces this time but we plan to use other

cues, such as objects or logos. We compared here word occurrence and tracking time in terms of a rougher segment occurrence, but the comparison of these heterogenous measurements remains open.

The design of our multimedia system completely falls into the *search* task as described by Brehmer *et al.*'s typology [15] (in their Fig. 1) consisting of two parameters: *search location* and *search target*, either being *known* or *unknown*. Each situation maps to a subtasks: *lookup, locate, browse* and *explore*. Usual search engines are often used for *lookup* tasks (with *location* and *target* both *known*). Video broadcasters such as Youtube link videos together to support *browsing* tasks (when the *location* is *known* but not the *target*). *Locating* tasks consists in *knowing* the *target* but not the *location*, made successful by keyword search. The visual cloud supports *exploration* and *browsing* tasks, our keyword search is too strict to provide proper *lookup* task support. It should be improved, together with video linking, to better support *lookup* and *locate* (beyond time location with the timeline).

One last important future work concerns *comparison* tasks. We currently refine information through visual cloud hovering, timeline browsing, and leapfrogging. However, beyond side-by-side comparison of two queries tabs, we do not have explored other means of comparison. This need quickly rises as we would like to compare periods of time.

Finally, we have presented a system designed for exploration of the NHK News 7 archive with a visual cloud, that improves from tag clouds in several ways. It takes roots in a multiplex network formulation, and uses group entanglement of search result to build a hierarchy with stable grouping, that results in a very fast and interactive drawing. The cloud is coordinated with search query, results, and timeline to allow further browsing, exploration, and query refinement. We illustrated our system with the case of Abe Shinzo and North Korea, studied the ability of our hierarchy to optimize group entanglement, and presented implementation and complexity.

Chapter 4

Visually tracking dynamic communities using Laputa's multiple coordinated views

This chapter reports on our work developed in close link with media researchers and media experts at INA and Sciences Po (Paris). It is driven by a high-level question: are potential bias induced from patterns in the way people get invited on TV and radio news events (when, on what occasion(s), in what context(s), with whom)? This question then induces more specific interrogations related to media exposure: are there co-invitation patterns? Do communities form in the media and how does the community structure relate to media visibility?

In order to detect and visualize community structure in the context of the French audiovisual news media landscape, we investigated data covering the French media over the 2011 - 2017 period. Assuming there are communities in this evolving ecosystem, how can they be characterized? How do they behave over time? Do known phenomena (related to social/political events or persons) act as underlying drivers of communities?

The analysis and visualization of communities in dynamic networks has received much attention these recent years [44] [11] [57]. Being able to find groups in data indeed is a central task in most taxonomies [1] [59], as it fundamentally relates to understanding structure and capturing insight in data.

Timestamped and even streamed data is now abundantly produced. While theory might be uncomfortable when defining what a community is in a time varying network, users usually have a quite clear idea about what types of groups they expect to discover and "see", or on the contrary what groups they expect not to see.

It is this situation we address in this chapter. We present an approach supporting the discovery and visual inspection of communities in evolving networks. The networks we consider however have several specificities challenging the actually available approaches for detecting *and* visualizing communities in dynamic networks. The persistence of links in time is a major assumption made in many studies [28] [5] [79]. They make sense in social networks (a new friendship relation usually remains for a significant time), but fail in other situations such as the one we consider here.

Our data is formed by looking at persons invited on TV and radio shows over several years, on a daily basis. Some programs are recurrent (daily, weekly or monthly), while others happen in the context of special events (sports, political elections, etc.). Persistence typically does not hold for co-invitation links. Indeed, although people get invited relatively often with the same people, they do not get invited with the exact same people, in a very stable manner, over a given period of time.

Among several motivating questions, our users wish to establish whether invitation patterns induce the same people to be repeatedly invited in the media, and whether this takes different forms depending on the themes being discussed (politics, sports, arts), whether one can see these patterns form in time, etc.

4.1 Domain questions and tasks specifications

This section lists tasks that were identified as key to answer experts' questions. Expert at INA somehow form the hypothesis that actors of the medias organise into communities: some actors only get invited in some TV or radio shows, only a few radio stations or TV channels have a voluntarily open policy and invite actors from all political sides; not all TV channel cover sports, etc. This latent hypothesis is crucial in our work and motivates the community-centered tasks we detail below.

Conversely, the behaviour of these communities over time is much less well known.

The first set of tasks is more "canonical" and focuses on finding structural and attribute features in the data. The second, user-centered, group of tasks was elaborated in close collaboration with field experts.

Each task is presented by giving the high-level domain question that underlies it; the question and tasks are then refined into lower level tasks following a methodology from [14] breaking down tasks into complementary why/what/how dimensions.

4.1.1 Task 1 - Communities, are you there?

Task group 1 gathers data-centered tasks and first looks at the very existence of communities (subtask 1.1): are there any? Can we detect them in the co-invitation network? Once we have an answer (positive, as suspected) to this question, we may form strategies to investigate them, how they can be characterized (subtask 1.2), and how they evolve in time (subtask 1.3).

Subtask 1.1 - (WHY) Confirm the relevance of the co-invitation network

(HOW)	This question is investigated by building and laying out a network of actors linked to one another whenever they have co-participated to some TV or radio program.
(WHAT/IN)	The network is built using data covering all TV or radio programs ranging from 2010 to 2016, listing all guests.
(WHAT/OUT)	Coordinated views including a node-link diagram and communities (convex hulls) together with a Sankey diagram showing how communities behave along time.
(PROCESS)	Node/edge filtering, clustering, graph layout, clustering on iterative overlapping sliding windows.

Assuming communities indeed can be mined, the next step is to investigate whether communities show specificities captured by attributes.

Subtask 1.2 - (WHY) Inspect/Identify community profiles

(HOW)	Starting from available metadata, compute statistics, look at value distribution, mean value, min/max values over time, histograms, etc.
(WHAT/IN)	Map TV/radio programs onto edges linking actors actors (media channels, media owners, presenters, ...); use individual attributes (gender, job types).
(WHAT/OUT)	Contextual information on community members inducing characteristics on the community itself.
(PROCESS)	Computing statistics, trends analysis on communities.

Media communities are, by essence, volatile. That is, with the exception of the news magazine presenters, the presence of guests mainly depends on what happened

in the last 24h before the show. Some news events are expected and planned, such as elections or festivals; others are unexpected, like earthquakes or crimes. These differences induce contrast between how communities evolve in time.

Subtask 1.3 - (WHY) Examine communities temporal behavior

(HOW)	Show how measures evolve in time such as community cohesion and the density of activity (intra-community connections) along time.
(WHAT/IN)	Any user selected community or group of actors.
(WHAT/OUT)	Timelines describing the selected community behavior.
(PROCESS)	Community activity index (custom statistics, see section 2.5), community cohesion measure inspired from [29].

4.1.2 Task 2 - Communities versus media events

By contrast, this task group is user-centered and focuses more on "facts", that is fact based explanation for the existence of communities, or of their evolution through time.

Subtask 2.1 - (WHY) Experts suspect communities emerge from known media events (e.g., Has the creation of the BeIN Sport channel in 2012 impacted the media communities around football?)

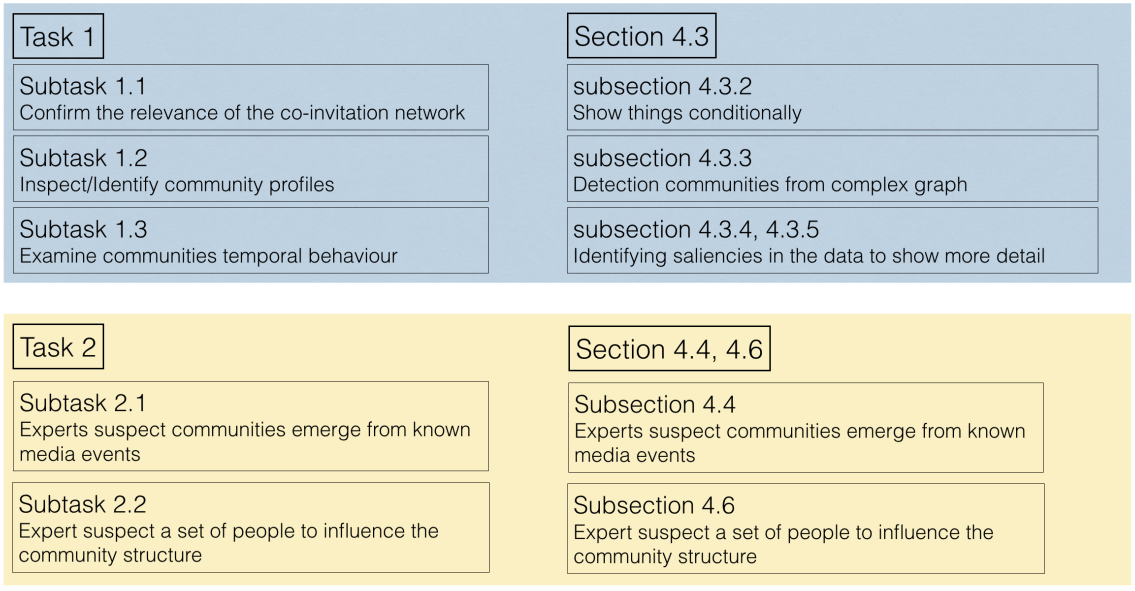
(HOW)	Drive the search from factual data (dates, topics, actors, ...)
(WHAT/IN)	Text query or selection of a visual entity (or both) to capture context.
(WHAT/OUT)	Display filtered node-link and/or Sankey diagram(s) corresponding to query/context; optionally run animation.
(PROCESS)	Filter network elements, filter timeline, trigger animation.

Similarly, an exploratory scenario may rely on a selection of a group of people that are suspected to belong to a same community.

Subtask 2.2 - (WHY) Expert suspect a set of people to influence the community structure (e.g. Can this be observed around the rise of media interest for Macron?)

(HOW)	Input people's names in textual query search engine.
(WHAT/IN)	Textual query (or boolean checkboxes).
(WHAT/OUT)	Display the induced ego-centered network(s) of selected people, together with contextual information (attributes).
(PROCESS)	Input/filter network elements, filter timeline, trigger layout and/or animation.

Fig. 4.1 The figures shows the chapter corresponding to each task

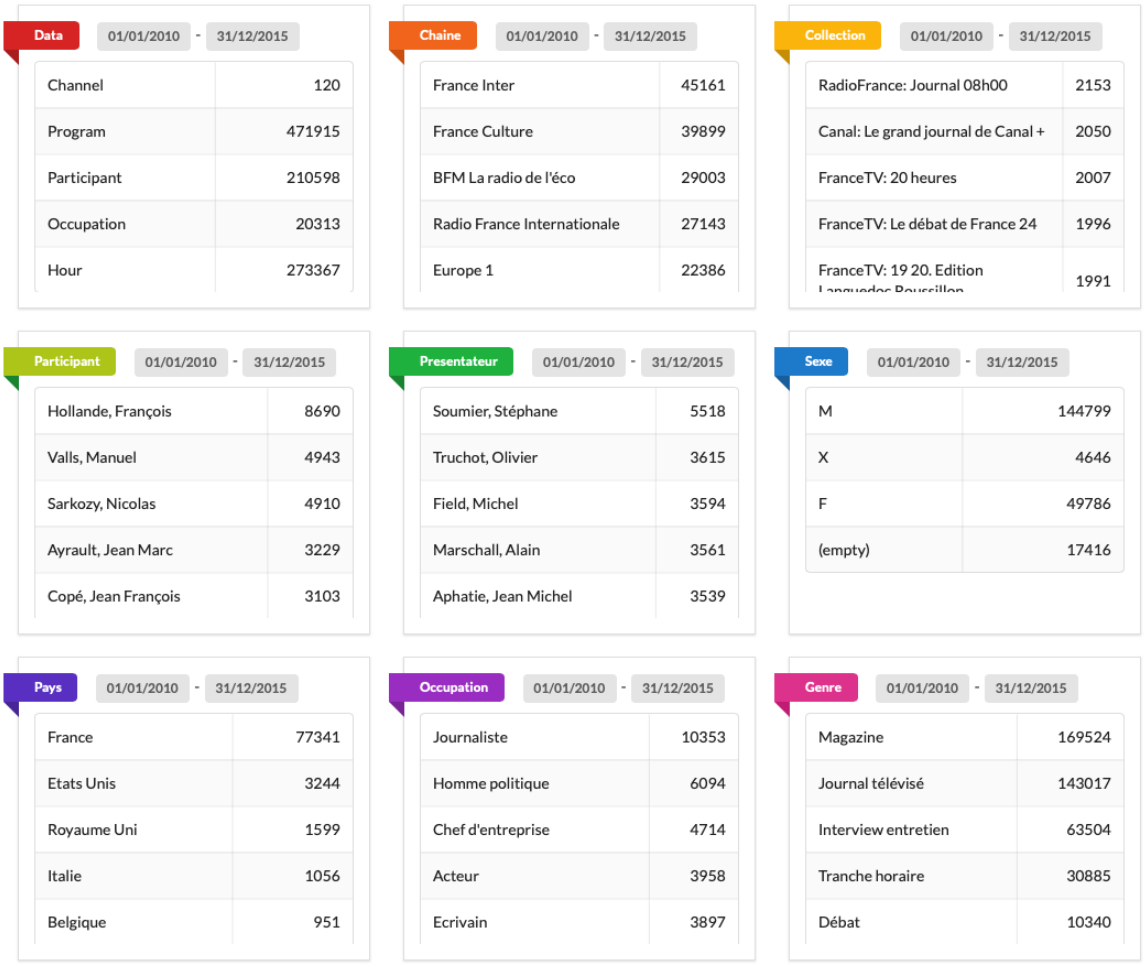


4.2 Data model

Our data consists of information on 471,915 TV/radio programs from France's 120 television/radio channels from 2010 to 2015, it involved 210,598 guests. Figure 4.2 display the composition of our data.

Although the notion of a community refers to a clear concept in sociology [77] [51], we shall refer to a community *in a graph* in a computational sense. We thus need to

Fig. 4.2 Two nodes are connected,



define *what* graph we are considering. In our context the graph $G = (V, E)$ we consider is formed of persons (nodes) that link whenever they are co-invited to a TV or radio program.

Let us denote by $\mathcal{P} = \{p_1, p_2, ...\}$ the set of media programs that took place over a time period. As figure 4.4 shows, since a person gets invited to numerous programs, each edge $e \in E$ can be mapped to a subset of programs $\omega(e)$ through a mapping $\omega : E \to 2^{\mathcal{P}}$. Similarly, given two persons $u, v \in V$, we write $\omega(u, v)$ to denote the set of programs they co-paticipated to (if any). We also will need a map $\sigma : \mathcal{P} \to 2^V$, mapping program p to its subset of participants $\sigma(p) = \{v_i, v_j...\} \subset V$. Now, each program is broadcasted at a given date and time t. We write $\tau(p)$ to denote the (start) date/time of program p.

Fig. 4.3 The co-participants network construction steps, (a) is the initial storage form of the data, (b) establishes the relationship between the program and the participants, (c) is the network composed of all the participants, they are connected when they are participate in a same program.

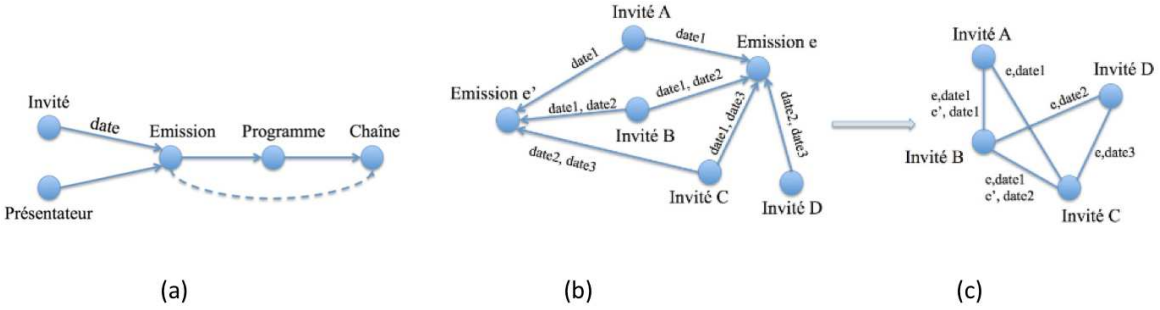
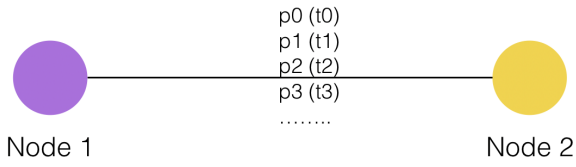


Fig. 4.4 each edgee can be mappedto a subset of programs, each program is broadcasted at a given date and time.



4.3 Statistics, visual encodings, and interaction

Following Munzner's nested model to design visualizations, we present here the rationale behind the Laputa framework that was designed to support the tasks listed in the previous section.

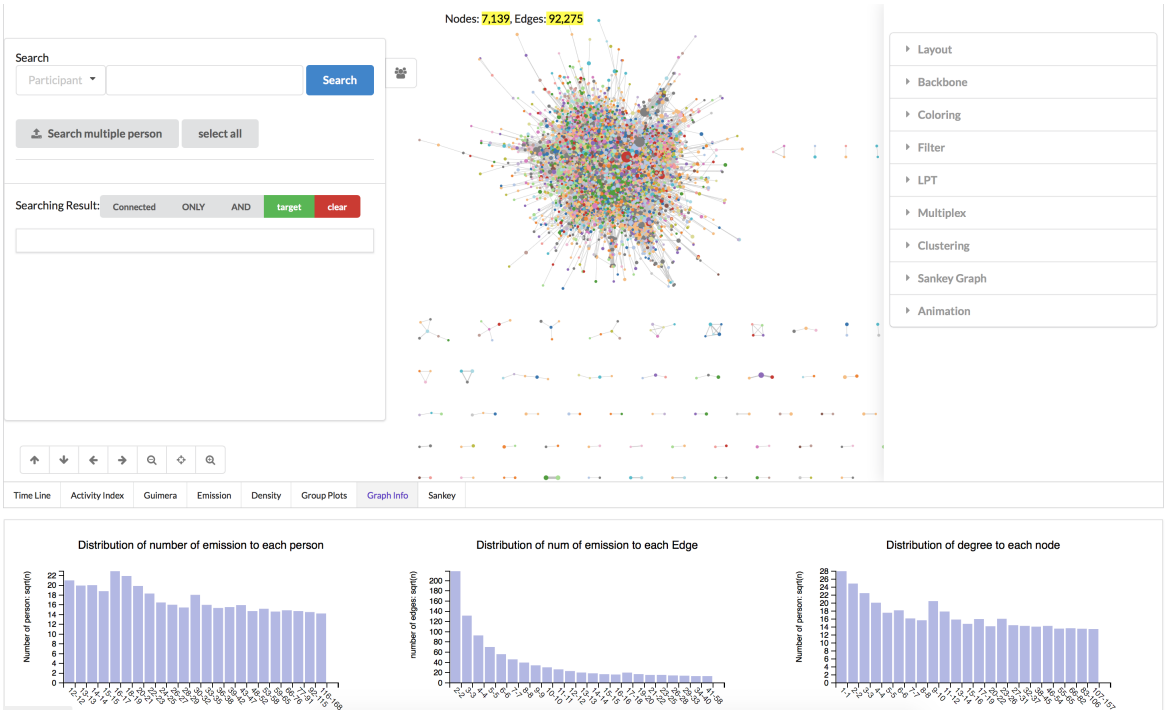
4.3.1 Overviewing the data, communities at large

The framework classically supports the first step of Shneiderman's mantra "Overview first, zoom and filter, then details-on-demand" [64]. An overview of the data expanding over the whole period is given using a force-directed layout algorithm FM^3[30].

This opening overview supports Task 1.1, at least partially, since it allows the user to examine the type of network co-invitations induce.

Indeed, in most cases force-directed algorithms naturally group nodes into *communities* that are visually separated (except in cases where the data becomes too dense

Fig. 4.5 The overview of our framework. The left part is the search function panel that can be toggled, the middle is the display area of the whole data, the right side is the operation panel that can be toggled, and the below is the data distribution histogram of the overview data.



[27]). Incidentally, a community detection algorithm [13] is run and nodes are colored according to the community they belong to, thus reinforcing how node positions can be interpreted.

This node-link view is complemented with bar charts indicating how node attributes distribute (see lower part of Figure 4.5). The leftmost chart provides information on how many programs a person has participated to; the middle bar chart provides the same statistics on edges, that is, how many programs two persons (incident nodes) have co-participated to; while the rightmost chart sketches the degree distribution on nodes, that is the number of persons a given person $u \in V$ has co-participated with.

4.3.2 Conditionally showing elements

For complex networks and multi-layer networks, one of the challenges encountered in the process of visualization is that the amount of data is large and the visualization results are complex. In order to allow users to understand the data more deeply, an interactive function-complete visualization tool should satisfy the user filtering

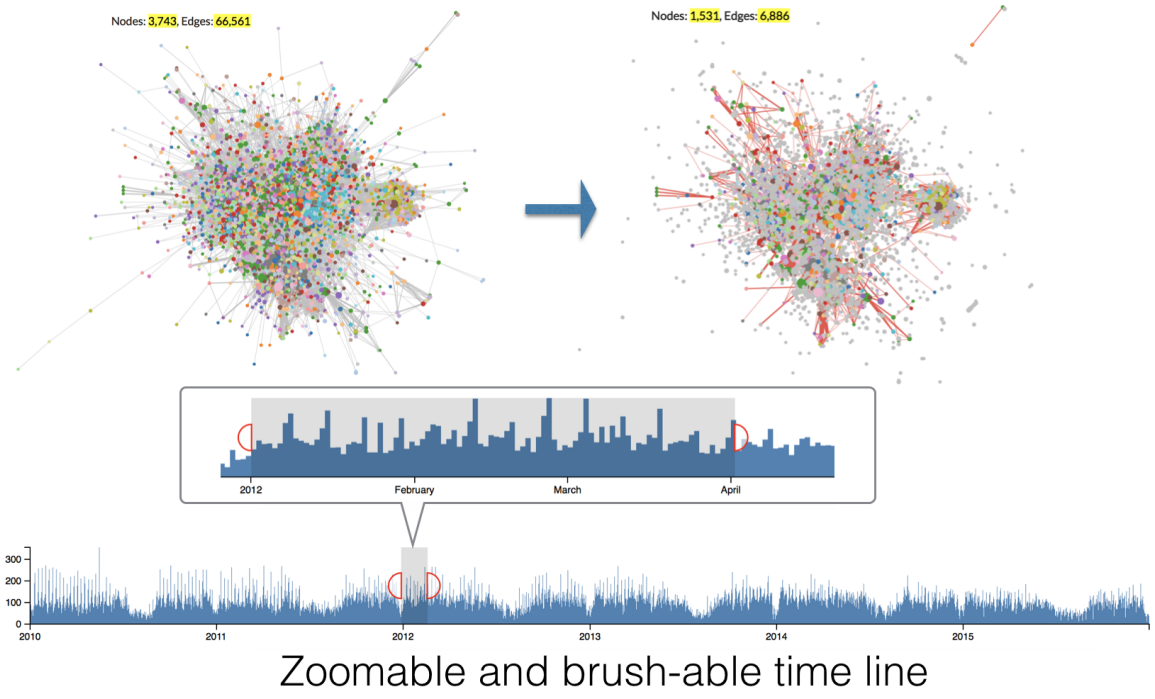
operation. In our visualization system, users can add a variety of different conditions to filter data.

Interactive timeline, filter graph by selecting time

This subsection describe how to solve the subtask 1.1 (Confirm the relevance of the co-invitation network). A time line provides an overview of the number of programs distribution over time, as Figure 4.6 shows. The blue line represents the number of programs chronologically. This representation already gives an overview of trends at a coarse level. For example, from the timeline below in Figure 4.6 we can clearly see that each year, during August and in the end of December, the number of programs is significantly less than other time periods, this is the change caused by the holiday.

Since we investigate large archives (over the course of years) but programs happen on a very fine grain level (every hour), we design our timeline to be extensible. As illustrated in Figure 4.6, users can zoom in to the smallest time interval. The timeline is also brushable and users may brush to select a period of time. This will filter the network view to only display the subgraph during that period. This brush may be used to define an aggregation time frame, so the dynamic graph may be *played* over time (given a tunable time-delta), bringing users another perspective of the dynamics of the network.

Fig. 4.6 This figure shows that users can filter graphs from the timeline. The y-axis represents the number of programs. Because the overall time period is long, we added a zoom function to the timeline, and the user can zoom out to a smaller time period to observe and select as needed. When the user selects a time period from the timeline, the graph displays the subgraph within the selected time period, and the edges within the time period are displayed in red.



Interactive histograms, filter graph by selecting properties

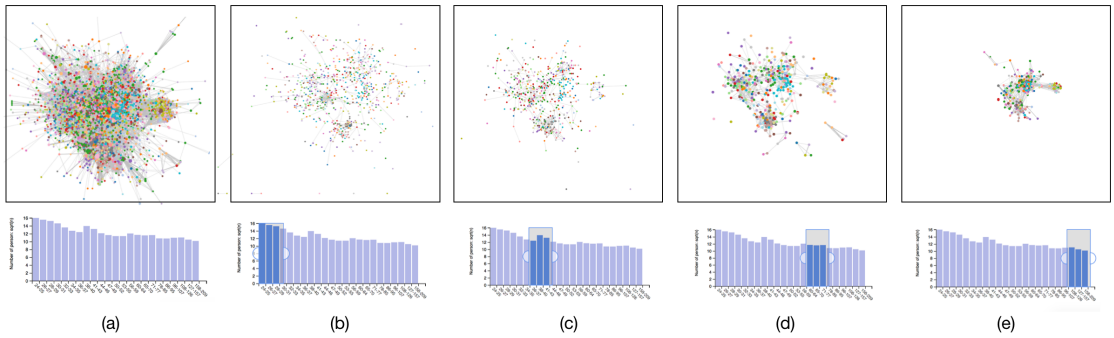
Overviewing several statistics of the graph is often necessary for a better understanding of the data. We offer histogram/bar chart views to display these statistics. These statistics are traditionally properties of nodes such as the degree, weight, of edges such as, weight, or computed statistics such as number of programs node involved.

In order to understand inner characteristics of the graph or relationships between different attributes, users may select interesting intervals from the bar charts by clicking bins or brushing. The corresponding subgraph will be filtered out in the global view, as shown 4.6.

As shown in Figure 4.7, users can select the different weight (the number of programs guest participating in) intervals to observe these correspond to different parts in the original graph. When the user selects the nodes with higher weight value, the

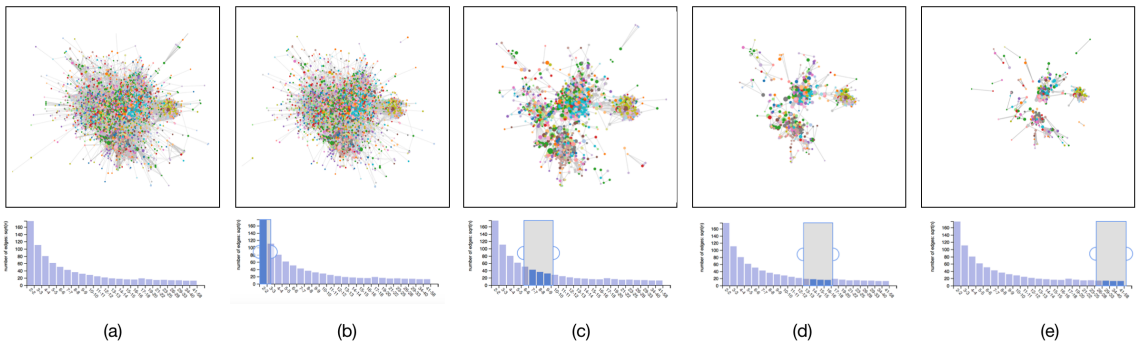
corresponding subgraph has a closer internal connection. On the opposite, the lower the weight value, the more dispersed the nodes are distributed.

Fig. 4.7 The histogram shows the distribution of nodes' weight (the number of programs guest participating in). The user can also select a section from the hitogram and then filter out the subgraph.



As shown in the figure 4.8, users can similarly browse through different weight of links (number of programs they involved together) intervals and visually observe which nodes are closely connected. In the co-participant data, link weights are obviously not evenly distributed. Similar to a power law, most of the weight of links values are relatively small.

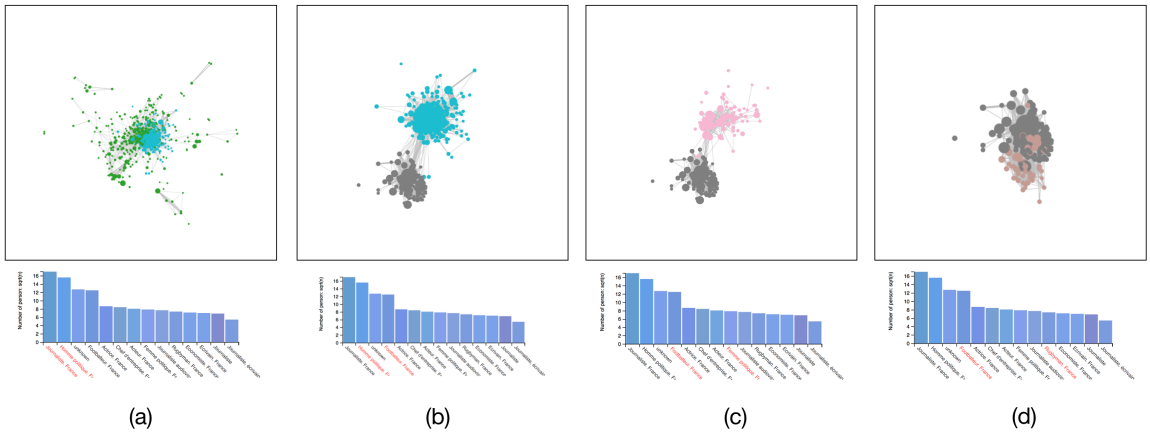
Fig. 4.8 Different histograms of graph properties allow the user to choose. The histogram in this figure is the distribution of weight of the edges (number of programs two guest participants together).



Categorial information also bear its importance. As a significant feature of co-participation, participants' occupation may explain the relationships in densities.

Interesting sub-graphs may be composed of different properties and investigated by selecting different occupation values in the bar chart. For example, as shown in Figure 4.9, *Journalists* are often connected with *Male politicians*. There is also a lot of connections between *Football players* and *Rugby players*. In addition, the relationship between *Male politicians* and *sports practitioners* is more frequent than that of *Female politicians* and *sports practitioners*.

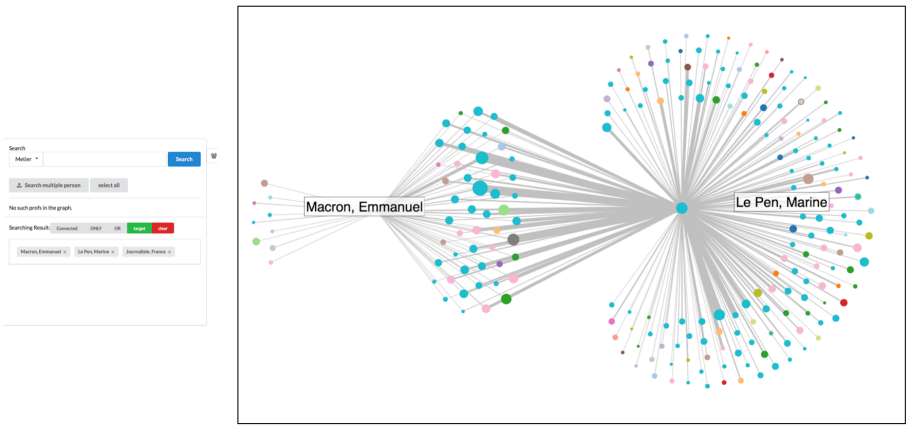
Fig. 4.9 The histogram of nodes' occupation distribution, user could select subgraphs consists by sertain occupations.



By selecting the properties, user could choose to filter out a certain type of sub-graph that meets the feature requirements, but it can hardly be specific to one person and/or the nodes connected to it. To this end, we have added an advanced search function to the framework. Users may search for one certain type of nodes/links or more, and also get the nodes connected to the search results.

For example, as figure 4.10, shows search for a subgraph composed of all journalists who have participated in programs with French politicians Emmanuel Macron and Marie Le Pen. The search function returns accurately the sub-graph as queried by the user, which may better fit further research needs eliminating unnecessary noise.

Fig. 4.10 An example of subgraph by searching from certain nodes.



Selection of different intervals over different feature distributions may be combined and selected at the same time. For example, we may select a sub-graph composed of co-participant that co-participated more than 100 times over the whole archive, and of which one of the participant must be a *Male politician*. Manipulation of attribute profiles makes it very convenient to filter and handle only subgraphs of interest.

4.3.3 Detecting communities from complex graph

On the complexity of the data

Our co-participants network is a graph built from six years of TV/radio programs, which is obviously a dynamic graph with many timestamps. However, due to the particularity of graph, the traditional dynamic communities detection algorithm cannot be effectively used in this graph. This specificity is reflected in:

- The complexity in time, because the start time of each program is accurate to the second, so the time-stamp of the network is extremely large throughout the all time period.

- The complexity in degree. The co-participant network is composed of cliques which makes the average degree of the graph high.

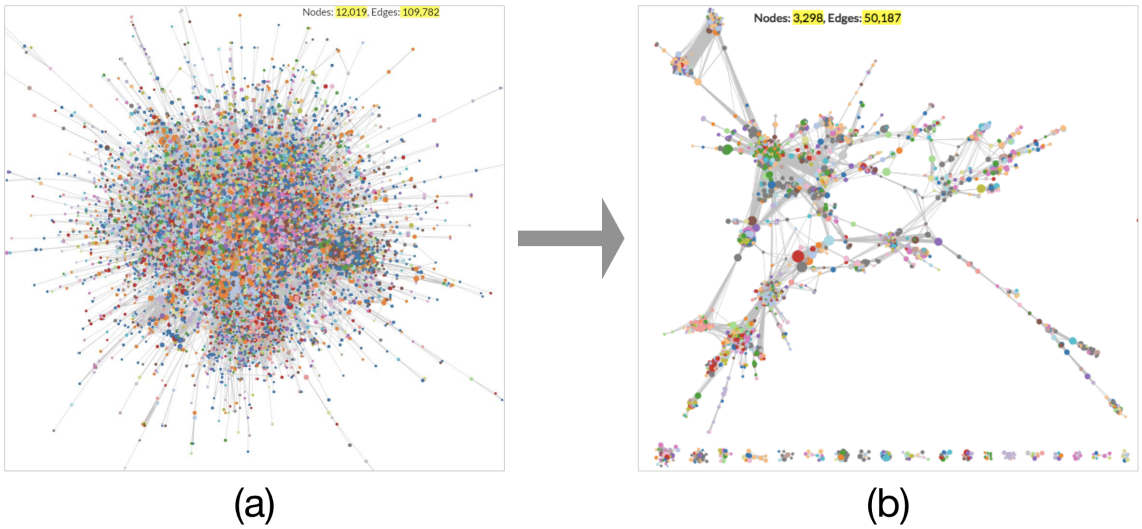
- The particularity of our data. Time stamp can be unified to the annual or monthly to reduce the complexity. However, the guests in the TV/radio program do not have the continuity of time. A large number of people only appear several times in certain time. They appear in a short time interval, but throughout the

time period, there are very few occurrences. Some participants participate in the program with a long interval and there is no continuity.

For the above reasons, we have used our own methods to detect communities from our data.

Using Simmelian backbone to delete redundant information and show a different arrangement

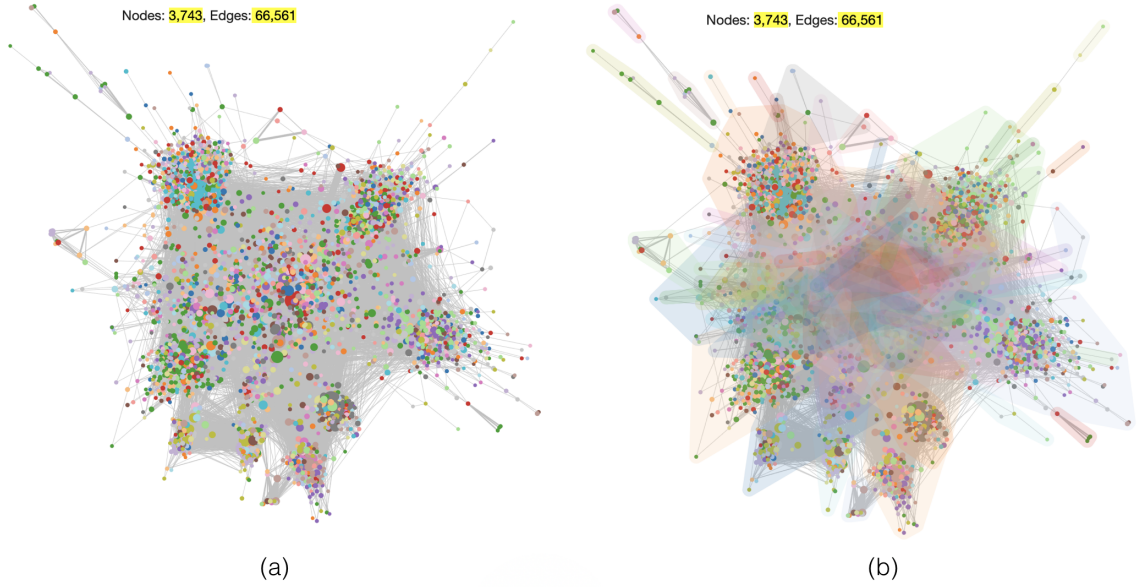
Fig. 4.11 (a) co-participation network built from 'Magazine' programs; (b) filtered network from (a) using Simmelian backbone.



Right after building the network from participation data, its structure remains very complicated and hard to read due to the density of nodes and links. As shown in the figure 4.12 (a), it is difficult to intuitively observe the characteristics of the network. Extracting groups directly from such a complex network often returns unpractical results.

We study the distribution of attributes in the original network, and these attributes make a good support for analyzing network characteristics. Indeed, not only the network shows structure, but also each different node and link bear its own set of attributes or value. Our intuition is that attributes also play an important role in shaping the communities in the network. Each different community has its own proper distribution of corresponding attributes. In order to reduce the complexity of our network, we need to remove redundant information hence we propose the use of the Simmelian backbone extraction method [48].

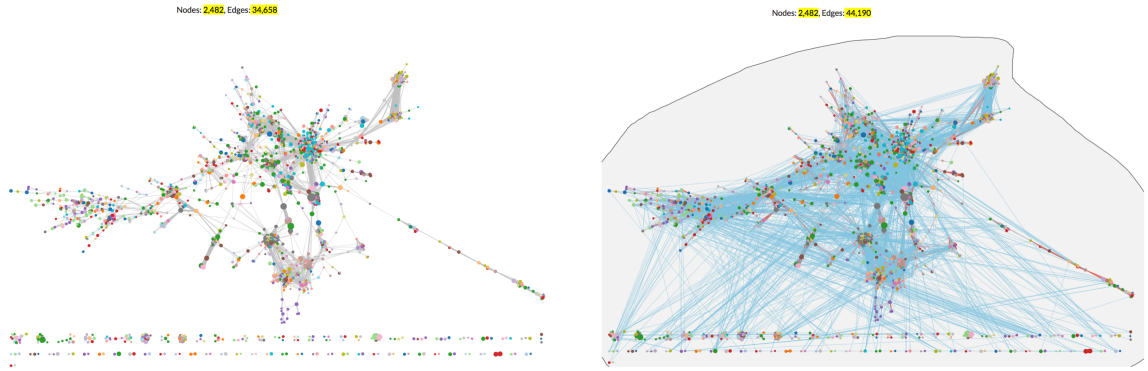
Fig. 4.12 (a) co-participation network built from 'Magazine' programs without using simmelian backbone; (b) using louvain modularity method directly to (a), nodes under same hull are from same group.



The Simmelian backbone method is based on the concept of triadic cohesion that is motivated by Simmel's concept. This method is used to identify the sentential relationships in network representing social interactions [48], described as follows: two people are *"Simmelian tied"* one to one another when they are reciprocally and strongly tied to each other, and if they are each reciprocally and strongly tied to at least on third party in common [36]. We modified the parameters of the Simmelian backbone method to apply it to our data, the methods are described in detail in the next chapter.

Since our network is a social interaction network, we applied the Simmelian method to our co-participation network. The resulting filtering looks promising. As illustrated in Figure 4.12, we can see the network before deleting the redundant information is deleted and the network obtained after using the Simmelian filtering with the parameters $m = 3$ and $n = 11$. As shown in Figure 4.12 (b) (the node color here illustrates the occupation of the guest), various communities of the network can be observed. Simmelian filtering is helpful to users so they may get closer to the communities. For further details, we prove in Chapter 4.5.1, from the network attribute distribution, that the Simmelian method removes unnecessary redundant data.

Fig. 4.13 The figures show the hidden edges, (a) is the graph user observerd, (b) shows the actual graph data, the blue edge is edge filtered by Simmelian method, we add it to the graph again, it do not participant in the display, but only in the actual calculation.



(a) (b)

In the original author's method, the number of traid is used as the basis for arranging links. At the same time, the author also pointed out that other weights can also be used as the basis for arranging links. In our data, program is the most important information, so we use the number of programs participating in the two participants connected as the weight of the link.

Supplement the deleted information

Using simmelian backbone makes communities more intuitive, but some connections between different communities are also removed, which affects the study of relationships between different communities. In order to specifically study the relationship between different communities, we re-add the links deleted in the simmelian backbone process to the graph. These links are hidden, do not participate in the layout calculation process, and will not make the visualization result appear complicated. As shown in the figure 4.13, (a) is the graph obtained after using the simmelian backbone, the graph has 34658 edges, and 4.13 (b) shows the all the links with hidden links of 4.13 (a). These blue links are information that is deleted, there about 10000 hidden links. These hidden links are recorded in subsequent calculations to supply more information.

Using traditional modularity methods to get communities from the filtered graph

We combine the various timestamps and divide the network with the traditional community detection algorithm to reduce the uncertainty caused by the difference in the time interval between different people participating in the program. After using the simmelian backbone to remove redundant information, we use the Louvin modularity algorithm for community partitioning, in addition while using Louvain modularity the weight of link is the number of programs. The latter is more efficient than the original, because it has been demonstrated that redundant data is deleted, so the quality of the community is guaranteed.

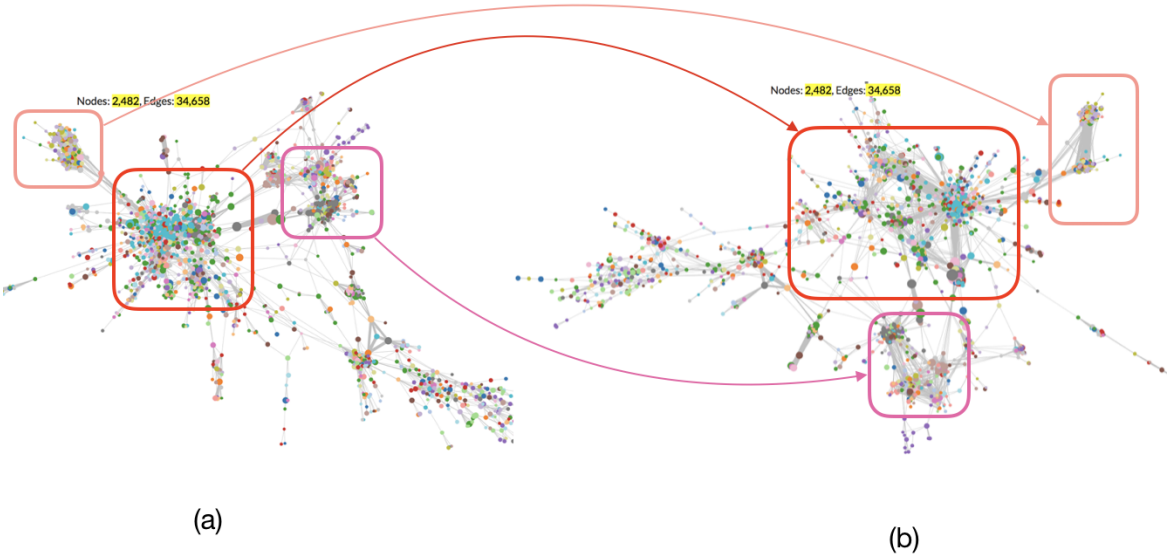
Optimizing communities visualization results

Although the use of simmelian backbone makes the graphics visually clearer, in the actual graph layout, there is still coverage between different communities, in order to visually make it easier to visually view the different communities in the graph, we modify the layout calculation steps:

1. Use communities detection algorithm to group the graph.
2. Reduce the edges between different communities to a certain amount.
3. Use the FM^3 force layout to calculate the layout of the graphs of the edges after filtering.
4. Add the edges that were filtered in the previous step to the graph.

Because the force layout is distributed according to the connection between the nodes, when the connection between different communities is reduced, the greater the repulsive force between them, the farther they are in space. In this way, without changing the graphics, the calculated layout can better disperse the nodes that are not belonging to different communities, and reduce the mutual coverage of the nodes, so that the user can better observe the internal and mutual relationship of the communities. As shown in figure 4.15 (a) shows the position directly calculated using the FM^3 algorithm, and the figure 4.15 (b) optimizes the position using the above method. It can be clearly observed that the structure of the (b) is clearer and there is less coverage. As figure 4.14 shows, through our steps, the complex central part of the figure 4.14 (a) is well distinguished in 4.14 (b).The hulls wrap the nodes in the same group, and we use different colors of hull to distinguish the different communities obtained by

Fig. 4.14 The comparison between (a) and (b) shows the difference between before and after the layout algorithm is modified. (a) is the force layout not be modified, The middle part is very dense. It can be clearly seen in the (b) that the dense portion of the pattern is dispersed.



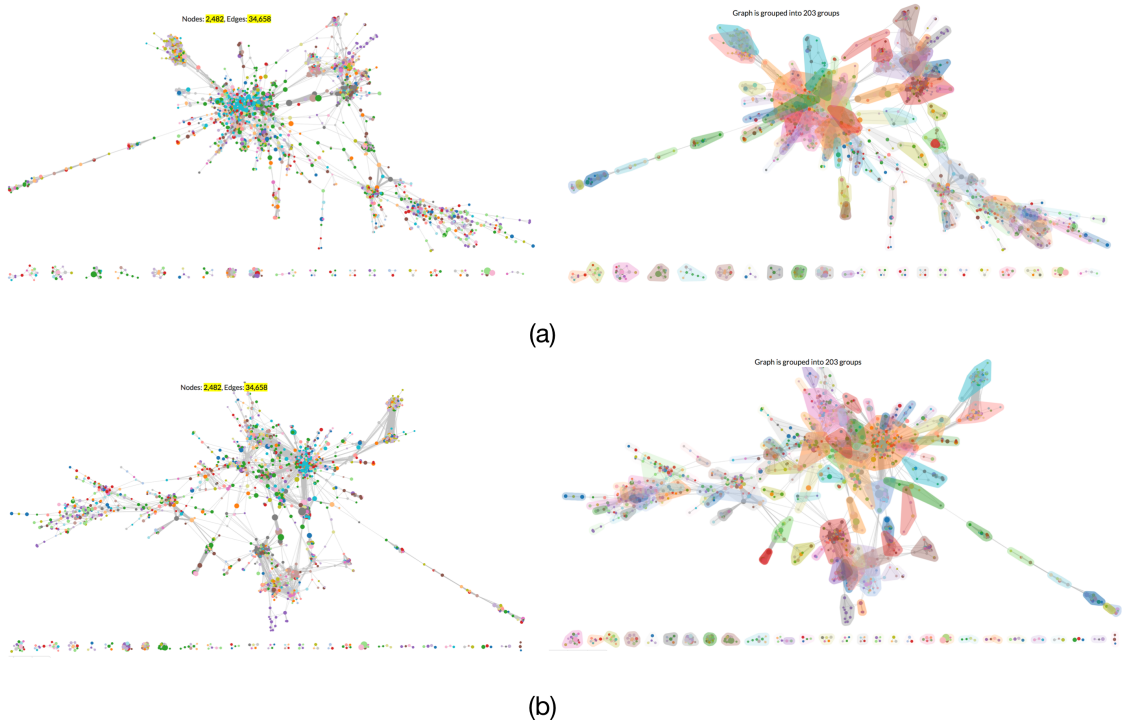
the algorithm. It can be seen that there are very few overlapping of the nodes from different communities. The different colors of hull not only prove the effectiveness of our modification of the force layout algorithm, but also help the user's intuitive understanding and selection of different communities.

4.3.4 Studying a community/an individual

Understand the composition of the community

The right panel has the accordion panel allows user to toggle the display of section of content of the features of the nodes. By clicking or draging the lasso to select community from the main graph, user can see the details of the internal members of the community in the accordion panel. We use the table to display information about the selected nodes, such as the name of each selected node in the co-participant network, the number of times the program is attended, the number of times the program participated with the internal members of the community, and the number of times the program was joined with other community members. For the co-participant network, we use labels to show the occupational distribution of the selected community,

Fig. 4.15 The different colors of the right side of the (a) represent different communities detected by louvain modularity algorithm. See that there is a lot of overlap between communities. The overlap can be clearly be seen to be much reduced in (b). The layout modification makes it easier to observe each community.



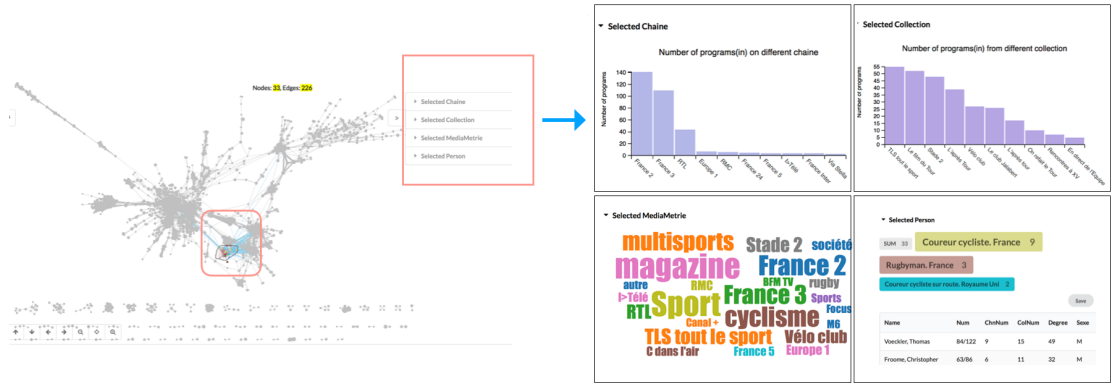
allowing the user to quickly understand each member of the selected community and its characteristics. And we used tag cloud as the information visualization to show the type of program, radio, and keywords that the members of the selected community participated in. And we have made statistics on the channels and types of programs that users participate in. Those figures allow the user to understand and explore the event that the community participates in.

Identifying key events associated with a community

As shown in the figure 4.17 (a), when user select a community or several nodes, the connected hidden edges represented in blue are displayed, the blue lines let users know the connection between the selected community and other external communities.

In order to let users know the event of the community, we designed a time-line to list the channel/collection that community members participated in. We use the month

Fig. 4.16 After the user selects a community from the node-link diagram, we provide a variety of views to let the user know the composition and activities of the selected community.



as the time period, each point in the plot represents the programs that participants participate in each time period, the blue dots represent the programs that the internal members of the community participate in, the red dots represent the programs that are participated by external community members. The size of the dot represents the number of programs. in the y-axis of the plot, the different channels/collection are ordered according to the total number of programs, and the more channels the community participates, the higher the position in the y-axis.

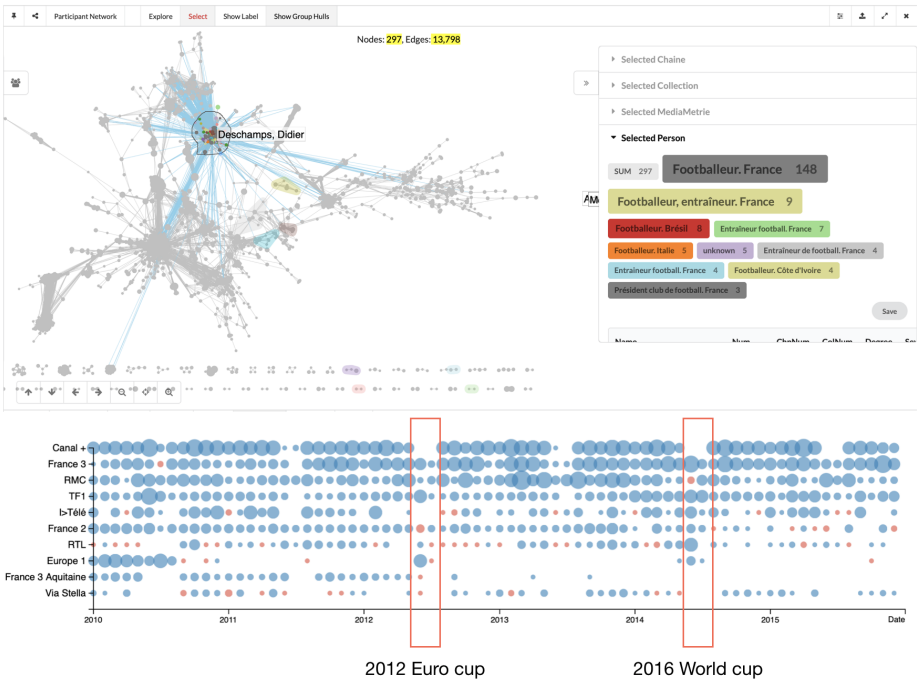
As figure 4.17 shows, when the user selects a community consists of people engaged in football, the main channel they participated in were Canal+ and France 3 and RMC, but during the European Cup in 2012 and world cup in 2014, their activities did not appear in Canal+, but in TF1 and Europe 1. This is because in the 2012 European Cup and the 2014 World Cup, Canal+ did not have the copyright, while TF1 and Europe 1 had the copyright, and the guests went to these channels to participate in the program.

Those figures allows the user to understand and explore the event which the community participant.

4.3.5 Identifying saliencies in the data to show more detail

Statistics usually are computed in order to get a sense on common trends, and to locate outlier individuals with respect with these trends. This section presents the

Fig. 4.17 After the user selects a community from the node-link diagram, we provide a variety of views to let the user know the composition and activities of the selected community.



various statistics we used, some of which that were specifically designed, in order to accomplish this in order to support the aforementioned tasks.

After we get the global communities, we define multiple indexes to study the activity of community and relationship between different communities. It is worth mentioning, all the index is calculated in the static filtered graph G'_l, for a community C_i the index is changing over time.

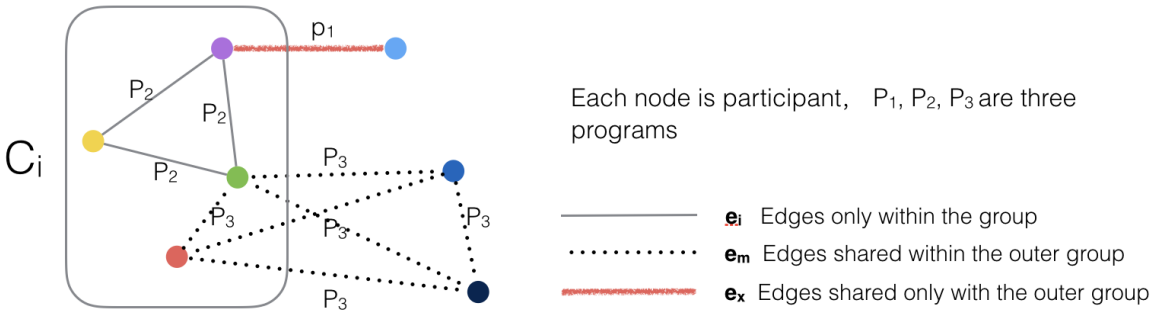
Formally, we define $\mathcal{P} = \{p_1, p_2, ...\}$ as the set of TV/radio programs, we define our co-invited network $G = (V, E, \omega)$, $\omega : E \to \mathcal{P}$. where V represents the set of participants (vertices), with $|V| = n$, n is the sum number of all participants. The edges $\{u, v\}$ exist when u and v are invited in at least one same program p. $\omega(u, v) = \{p_i, p_j, ...\}$ are the programs u and v co-invited together.

Because for each program we know its start time. For convenience, we define $\tau(p)$ as the start time of program p, define $\sigma : P \to V, \sigma(p) = \{v_i, v_j...\}$ as the set of participants in the program p.

For each community C, we know all the programs they co-invited together, we define $P = \omega(C) = \{p_x, p_y...\}$ are the programs which have at least one participant from community C, Obviously $\forall v, v \in \sigma(P_i)$ we can not say $v \in C_i$, $\sigma(P_i)$ is the set of

all the participants in the programs which have at least one participant from community C_i, that means in the programs P_i which have invited the members from community C_i may also have invited the participants from other communities, $\omega(C_i, C_j)$ is the set of programs which have participants both from community C_i and C_j. Therefore we could find three different types of interaction for the members in a community, as figure 4.18 shows:

Fig. 4.18 Three different types of contacts between members.



- if $\forall p, p \in P_i \wedge \sigma(p) \subseteq C_i$, that means all the participants in the program p are from the community C_i, we mark those programs as internal programs P^{in}, such as program p_2 in figure 4.18.
- if $\exists v, v \in \sigma(p) \wedge v \notin C_i \wedge |\sigma(p) \cap C_i| > 1$, the participants in program p are not all from community C_i, but at least two participants are from community C_i, we mark those programs as mix programs P^{mix}. Such as program p_3 in figure 4.18.
- another type of interaction is $\exists v, v \in \sigma(p) \wedge v \notin C_i \wedge |\sigma(p) \cap C_i| = 1$, that means only one participant from community C_i is invited with people from other communities, there is no interaction in the community C_i, we mark those programs as external programs P^{ex}.

Fig. 4.19 An example shows how indexes lines work. When user slects a community, the index of the community will be displayed by time line charts, showing how the community changes over time.

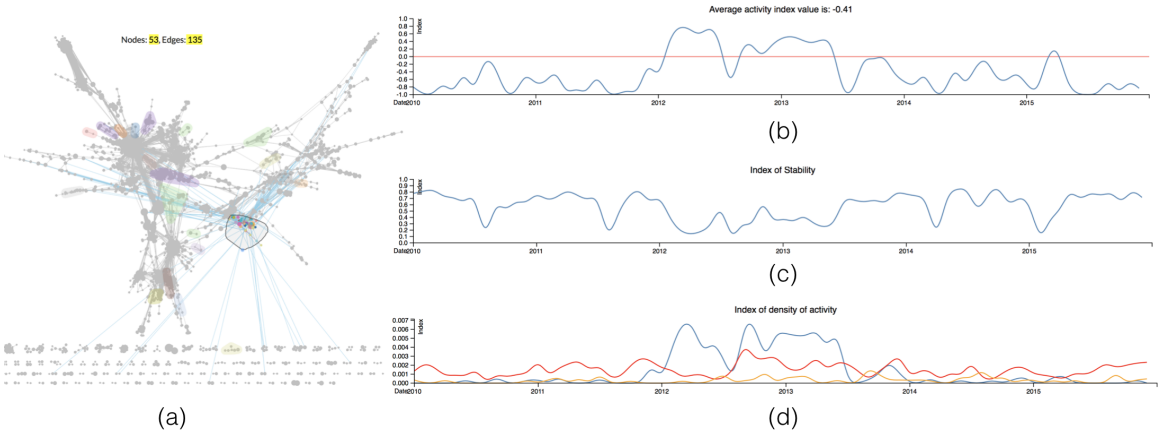


Figure 4.19 (d) shows the changes of the number of programs for a community over time. The blue line is the change in the number of programs that only internal members of the community participate in, that is $|P^{in}|$. The yellow line is the distribution of $|P^{mix}|$ over time. The red line is the change in $|P^{ex}|$ over time. Obviously, a good community partition is to maximize the $|P^{in}|$ of each group, with $|P^{ex}|$ being smaller.

Activity index

Activity index is used to judge whether there are lots activity between the members from same community over time. A good community division should be to group together participants who often participate in the program together, the number of programs co-invited members among different communities should be less. for a community C_i, during a time interval $[t_0, t_1]$, we define ι_i as activity index of community C_i during time interval $[t_0, t_1]$, where

$$\iota_i = \frac{|P_i^{in}| - |P_i^{ex}|}{|P_i^{in}| + |P_i^{mix}| + |P_i^{ex}|} \tag{4.1}$$

This value is between -1 and 1, if ι_i is -1, that means there is no interaction within the community C_i, all the members of C_i are invited alone with member from another community. During a time period when the members from community C_i are invited without members from any other communities, ι_i is 1.

Density index

TV/Radio shows are cyclical, with daily shows, weekly shows, and monthly shows. In these periodical programs, the participation frequency of participants is relatively stable, in order to compare and to study the percentage of participants in the community participating in programs at different time intervals within the participation in programs of all of the communitities at that time interval, we define the density index:

$$\delta_i = \frac{|P_i|}{|P|} \tag{4.2}$$

Through the density index, we can observe the event that is different from the overall trend and help users discover some special events.

Stability index

Stability index is used to analyze the relationship between different communities and determine whether connection between communities is consistent overtime. An unstable index means the selected community connect with different communities at different times. We define γ as the index of stability for the community C_i during time interval $[t_0, t_1]$.

$$\gamma_i = 1 - \sum_{j \in C} \frac{|P_{(j,i)}|}{|P_i|} \tag{4.3}$$

where P_i is the set of programs in which invited participants from community C_i, and $P_{(j,i)}$ is the set of programs in which invited members are both from community C_j and C_i.

If the stability index is 0, it means that the selected community C_i is only related to one community (including itself). If the community is connected to a number of different communities, it means that the members of the community C_i are not stable and stability index could be high.

4.4 Visually tracking communities' evolution

After efficient access to communities, we are more focused on the evolution of communities and looking for reasons for their evolution. In our framework, three different methods are provided to demonstrate the evolution of the dynamic graph. First, the most intuitive way to track the evolution of the community over time is to show the process of community change through the animation method. But animation only gives the user an impression of the change process and cannot quantify the specific

Fig. 4.20 Indexes used to help user observe the time period when the selected community is not active.

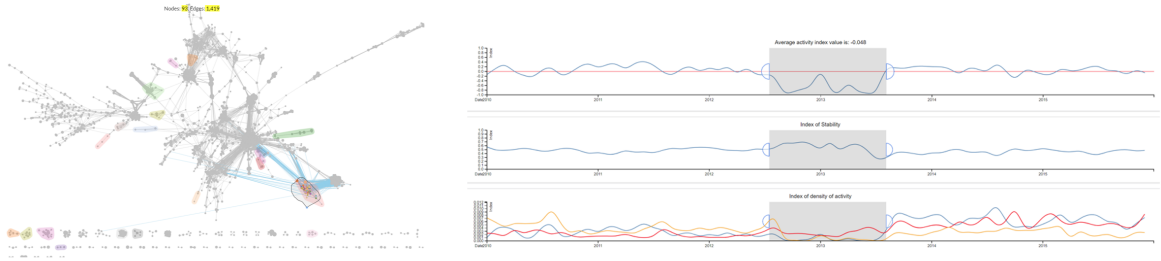


Fig. 4.21 Indexes used to help user observe the time period when the selected community is much active.

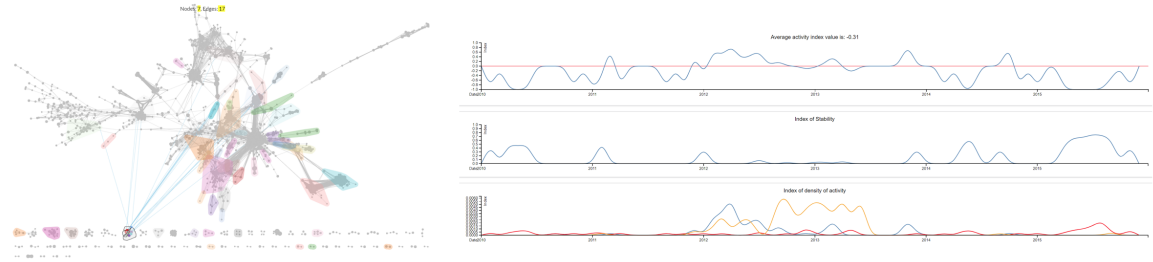
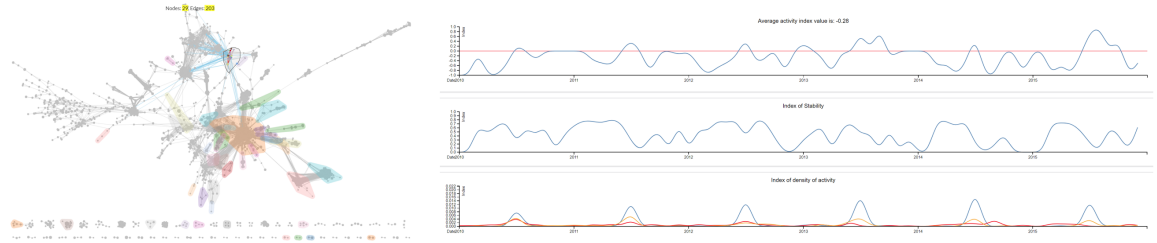


Fig. 4.22 Indexes used to help user observe the periodic activities of the selected community.



change process. So we list the subgraphs of different time periods, the evolution of the community is shown by comparing the position of each community in the subgraphs at different times, but this method can not observe the changes of all communities at the same time. So we introduce the sankey flow graph, and all the communities can be visually observed during the evolution process.

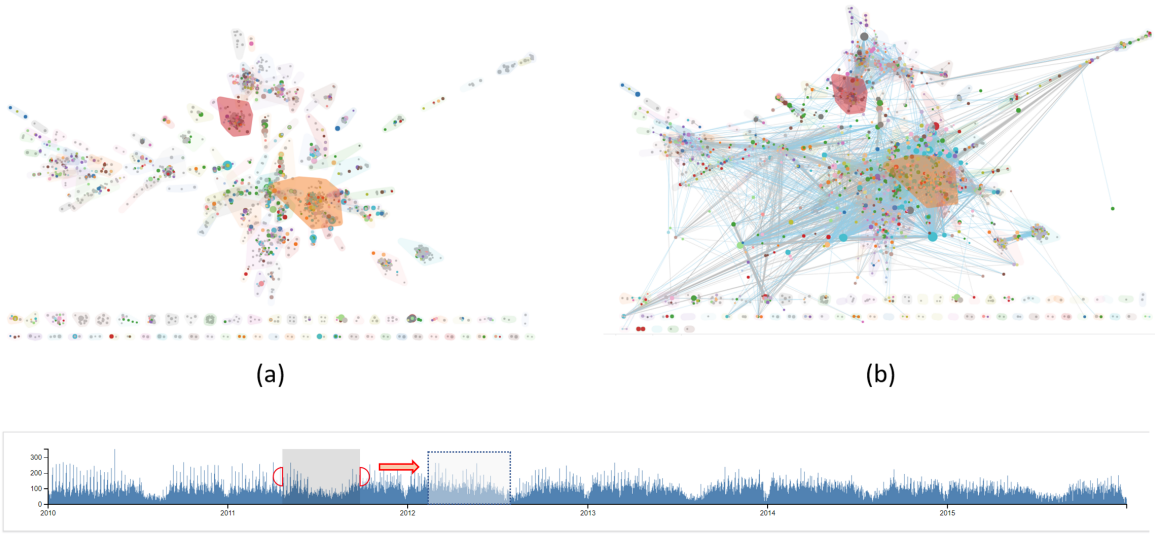
4.4.1 Animation

Inspired by the flight flow visualization ref, in this visualization, the running process and running track of each aircraft flight are shown in an animated form. So we want to show the change process and change track of nodes belonging to different communities at different times in an animated form. In the global graph, communities C is divided according to the total time T, and our modified force layout ensures that the nodes belonging to the same community are closer together in the position of the graph. For a certain time period (t_0, t_1), we group their corresponding sub-graph $G(t_0, t_1)$ to $C(t_0, t_1)$, and then in the next time period (t_1, t_2), we filter sub-graph $G(t_1, t_2)$, use the same community modularity method to group the graphs, get $C(t_1, t_2)$, for each community c in $C(t_1, t_2)$, we count in which community each node of c at the previous time $C(t_0, t_1)$, get the community in which the nodes in c are most mainly located in the (t_0, t_1) time period, we use the minimum change criteria, select the position of the nodes in the community $c(t_0, t_1)$ as the end position, and the nodes of $c(t_1, t_2)$ do not belong to $c(t_0, t_1)$ in $c(t1, t2)$ move dynamically to a position where $c(t_0, t_1)$ community is. The initial layout of the graph is calculated based on the total time. As time changes to (t_0, t_1), the nodes in the same community during the (t_0, t_1) time period are gathered together in the layout, and through the above method, the number of moving nodes is minimized.

In our framework, the user controls the operation of the animation by controlling the timeline. In animation mode, when user can select any time window through brushing on the timeline, the system selects the sub-graph in the corresponding time period, groups the graph with Louvain modularity, and then nodes in the same group move together in the graph. The time window selected by the user can be automatically moved on the timeline, and the nodes will automatically move to show the continuous evolution.

During the animation process of node movement, users can observe the evolution process of each community, including: fusion, splitting, disappearing, gradually expanding, and gradually shrinking. However, because there are many nodes in the graph, when all the nodes are moved, the overall visual effect is rather messy, and the user needs to pay more attention to observe. In order to be more visually convenient, user can also select a specific community and then observe its changes in all time periods, when user selects a certain community, only the nodes in the graph that are associated with the selected community will dynamically move, and other nodes will not change. We can observe how this community evolves with time, when a member of selected community joins another community, they move to another community

Fig. 4.23 Screenshot of the animation process. The nodes in the graph will move with the change of the time window on the timeline. The (a) figure is the process of all nodes cues. In order to make the moving process more obvious, the user can choose to display the links at the same time as shown in the figure (b).

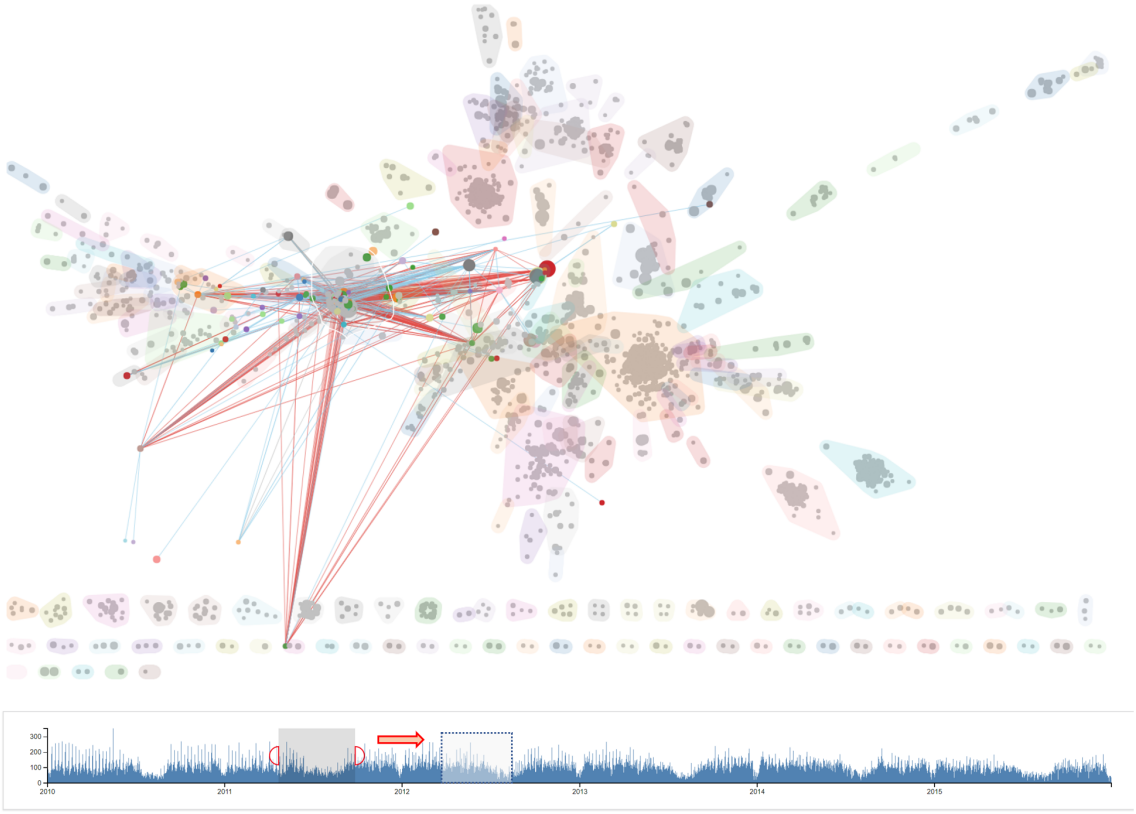


location. When other community members have more association with the selected community during a certain period of time, those nodes will move to the community. This smooth animation can visually demonstrate the process of community evolution, giving users a comprehensive impression, but can not quantify the user to accurately grasp the process of tracking evolution, for which we have segmented the graph in all time periods, subgraphs of different time periods are displayed. Users can track specific evolutionary processes.

4.4.2 Multiple views

Animation can visualize the evolution of communities, but it cannot simultaneously show the evolution of different time periods. To do this, we want to display different subgraphs simultaneously through multiple views, and provide methods for users to effectively compare the changes in the community between different subgraphs. To this end, users can select sub-graph from the global view, under which our framework than generates a new view window. The global view window and the sub-graph view windows are connected to each other.

Fig. 4.24 Screenshot of the animation process. Users can also select a community to observe the dynamic movement of all nodes associated with the community.



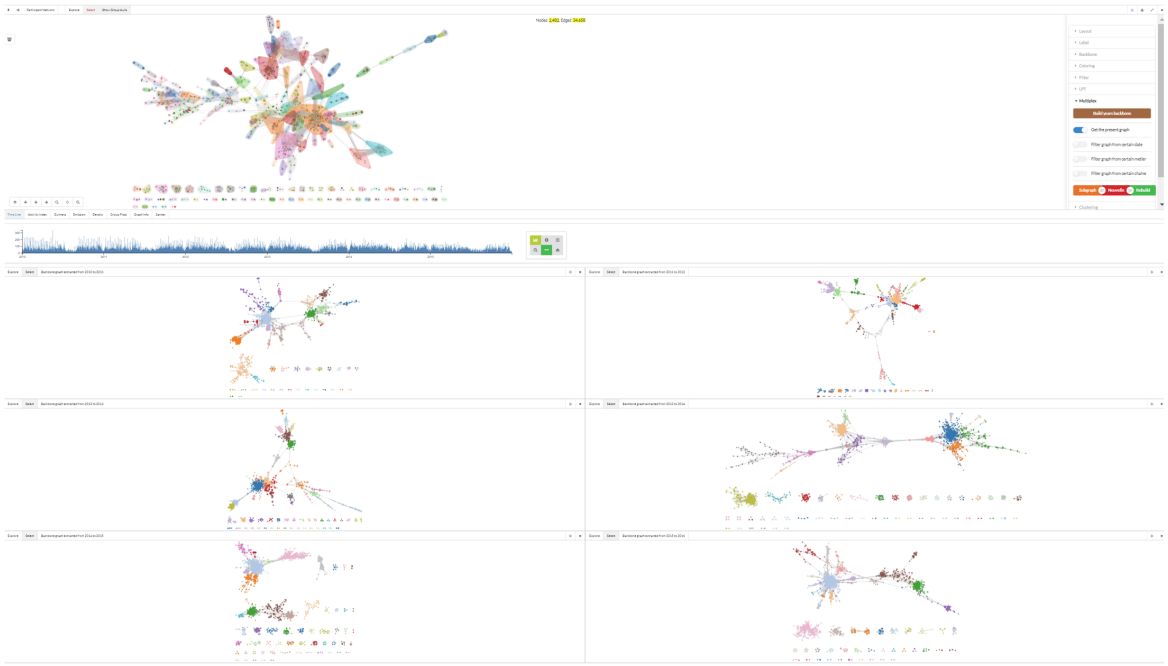
How to get sub-graphs from different time periods

In order to compare the evolution process of the community at different time periods, it is necessary to display the graph of different time periods at the same time. In our data, select one year as the time period and create a different subgraph $G(t_i, t_j)$ from the original network G. As the figure shows, however, the graph with the year as the time period is still quite complicated. In the global graph G, we use the Simmelian bakbone method to obtain the graph G_{sb} for deleting redundant information and get a clear graphical construct. Similarly, we also use the SB method in each subgraph. Obtain $G_{sb}(t_i, t_j)$, and then group $G_{sb}(t_i, t_j)$ using the Louvain modularity method.

The processing of each subgraph is the same as the processing of the initial total graph. In the global view, graph G is processed by the SB method and the Louvain modularity method.

$$G \rightarrow G_{sb} \rightarrow C_{sb} \tag{4.4}$$

Fig. 4.25 Screenshot of our framework, multiple views of graphs are shown at the same time.



In the subgraphs of different time periods, we also start from the initial graph G, and first obtain the subgraphs from different time periods. Then we deal with the same steps as the global view. That is,

$$G \to G(t_i, t_j) \to G_{sb}(t_i, t_j) \to C_{sb}(t_i, t_j) \tag{4.5}$$

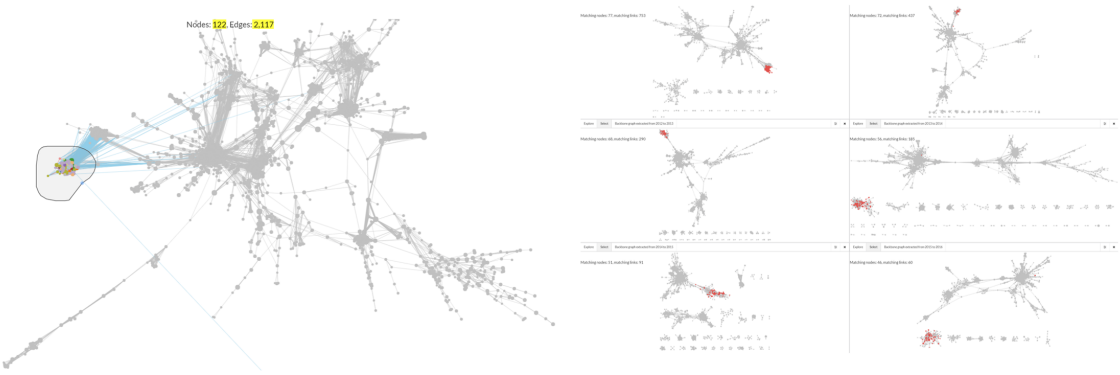
We could also use the following method to get subgraphs communities.

$$G \to G_{sb} \to G_{sb}(t_i, t_j) \to C_{sb}(t_i, t_j) \tag{4.6}$$

The results obtained by different methods are different. In our research we use method 4.5 to get subgraph communities. Subsequent chapters will specifically discuss the differences between the two.

Through the above steps, we divide the graph into subgraphs in different time periods, and obtain the community allocation in different time periods. The subgraph and the global graph are displayed together at the same time. The next step is how to compare the community from different time period.

Fig. 4.26 When user select a subgraph from gloabl view, the same nodes and links in different layer views will also be filtered out for display.



Looking for the evolution of the community between different view

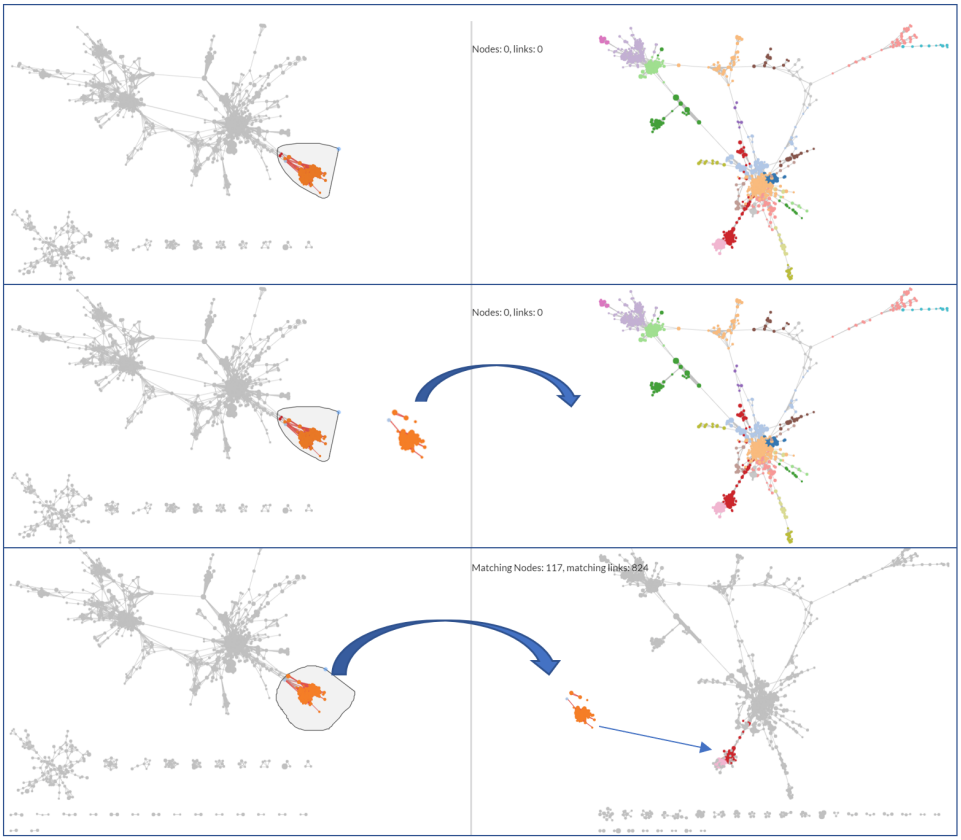
In our framework, each subgraph is coordinated with each other. When the user selects a community c from the global graph, the nodes in the different subgraph view that are the same as c can be displayed in bright red. The nodes different from c are grayed out, so that it is convenient to observe the members of community c from different time periods. Similarly, when user selects a community in the subgraphs, its members are presented in other sub-graph views and global graph. This allows you to observe the location of community $C(t_i, t_j)$ members in a certain period of time in all time periods and other time periods.

The advantage of showing subgraphs together at different time periods is that evolutions such as splitting, aggregation, etc. can be found. However, the disadvantage of this approach is that can only understand the convergence process of the community, but cannot track which community the specific nodes are in.

Drag-drop interaction In order to specifically track how the nodes in the community evolved over different time periods, we implement drag-drop interaction. The user can select any community $c(t_i, t_j)$ from subgrpah, and then drag it to other subgraph $G(t_x, t_y)$. The nodes and links in $c(t_i, t_j)$ will gradually move to the same nodes and links above $G(t_x, t_y)$, and the nodes and links of $c(t_i, t_j)$ that do not appear in $G(t_x, t_y)$ will disappear, nodes and links in $G(t_x, t_y)$ that don't match $c(t_i, t_j)$ will be grayed out. As figure 4.27 shows the process of drag-drop interaciton.

This interaction is implemented based on html's drag-drop interaction. So when we select from a sub-graph view, it creates a new layer on the original graph. When

Fig. 4.27 The process of drag-drop interaction. User select a subgraph from left view, then drag the selected subgraph to the right view, then drop the subgraph, the same nodes and links will gradually move to the matching location and then be highlighted.



the user selected a community, the newly created layer demonstrates the selected community,and as the user is dragging, the user is actually dragging the newly created upper layer . In the actual process, only the svg element of html can be dragged, and the canvas element does not have the dynamic effect of drag, so the newly created layer and graph are both on the svg element, after the drop event is fired, we also create a new layer on the dropped sub-graph view, copy the dropped community, then keep the same nodes and corresponding links, then add animation of nodes and links to move to the corresponding location. This interaction process and dynamic effect are not difficult to implement with the help of html.

The drag-drop interaction feature makes it easy to observe the evolution process, and can track which nodes are specifically changed to which community, and can map the same elements in different graphs one by one.

4.4.3 Sankey graph

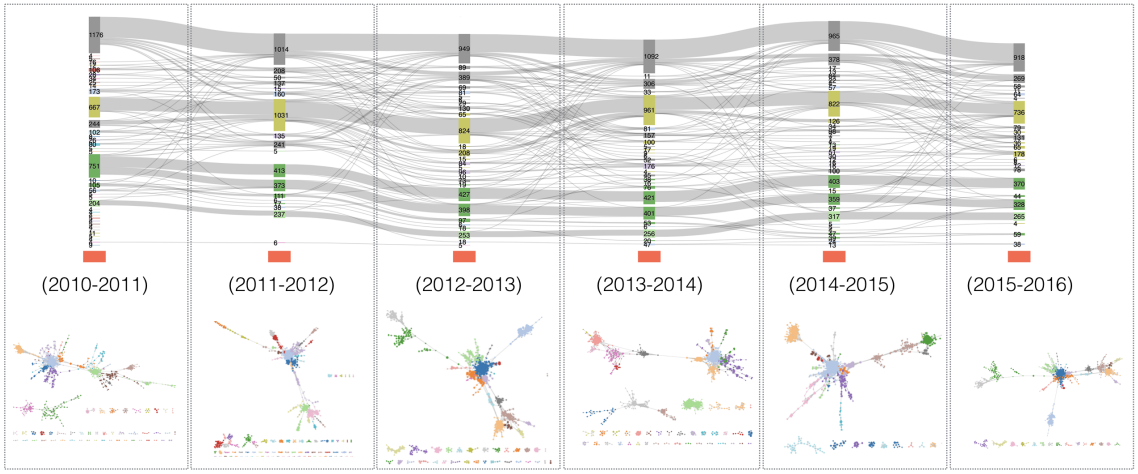
Using the global graph view, the graphs of all time periods are merged together. The user can group the graph of the whole time period and select the subgraph of a specific time period through the time-line to study, but the connections between the subgraphs in different time periods cant be reflected. For complex networks, we use the SB method to remove redundant information, making the graphics more intuitive and easy to analyze. Although the SB method deletes some unimportant nodes and links, it still causes the lack of information. So we conceived to study the evolution of the community without deleting the redundant information. When the redundant information is not deleted, we need to consider other visualization methods due to the node-link visual complexity.

$$G \longrightarrow G(t_i, t_j) \longrightarrow C(t_i, t_j)$$

To this end, we used a sankey graph to visualize the evolution of communities in graph at different time periods.

Sankey graph is a type of flow diagram, in which the width of the links is shown proportionally to the flow quantity. For our data, the sankey graph $G = (V, E)$, V is the communities detected from different times interval. Each band represents the community, the height of the bands are directly proportional to number of members of the community.

Fig. 4.28 An overview of sankey graph for 6 years.

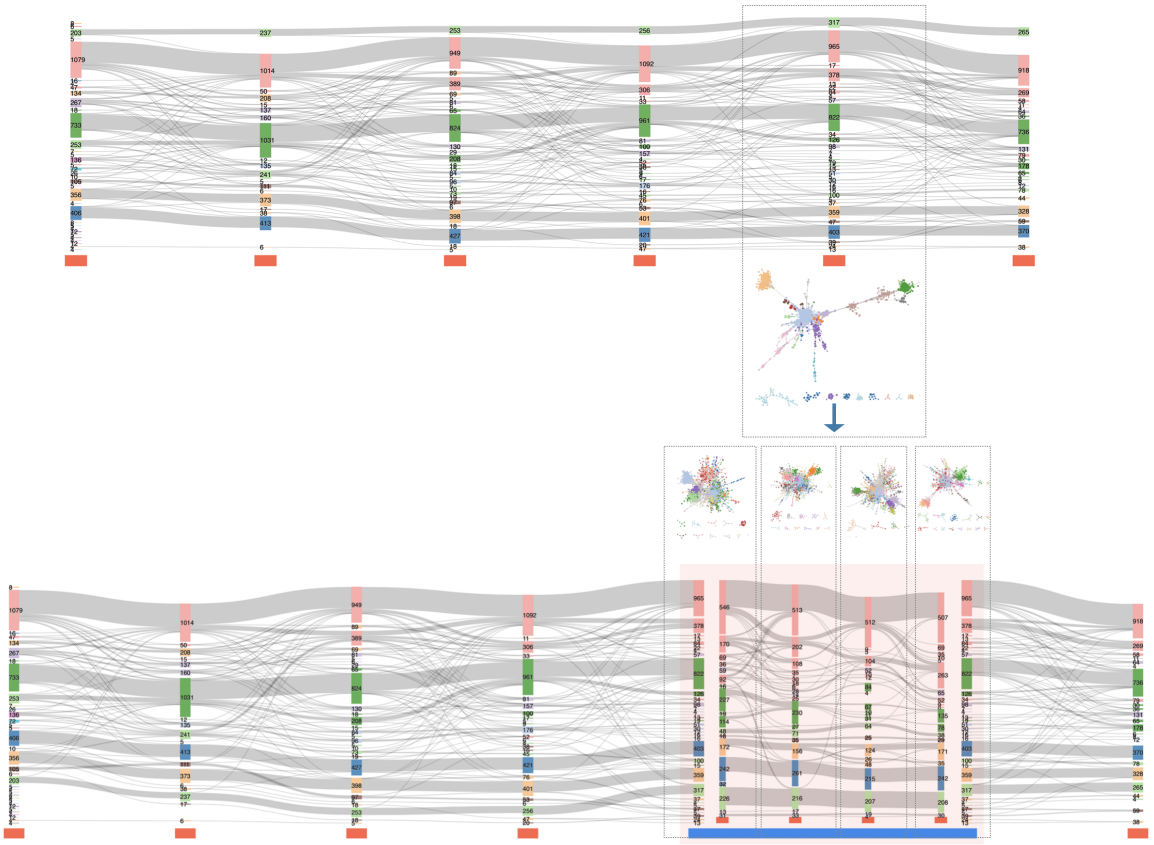


The Sankey graph is mainly to study the evolution of dynamic communities, because the mergering, separating, gradual disappearing, gradual increasing of the communities and so on an be observed at a glance through the sankey graph.

In our application, the sankey graph is made up of many rectangular and thick curves. As shown in the figure, each column rectangle represents the different communities in the sub-graph obtained in the same time period. The width of the rectangle is the same. The height of the rectangle represents the number of members in the communities. A thick curve joins communities with two adjacent time periods of the same member.

Unlike Equation 3.5, we did not use the sb method. In this case, there is no missing information in each time period. Although some nodes only exist for a short time, they are also displayed in sankey. As shown in the figure, sankey's rectangle has a section that grows up, representing the nodes that appear in that time period, and does not appear in the two adjacent time periods.

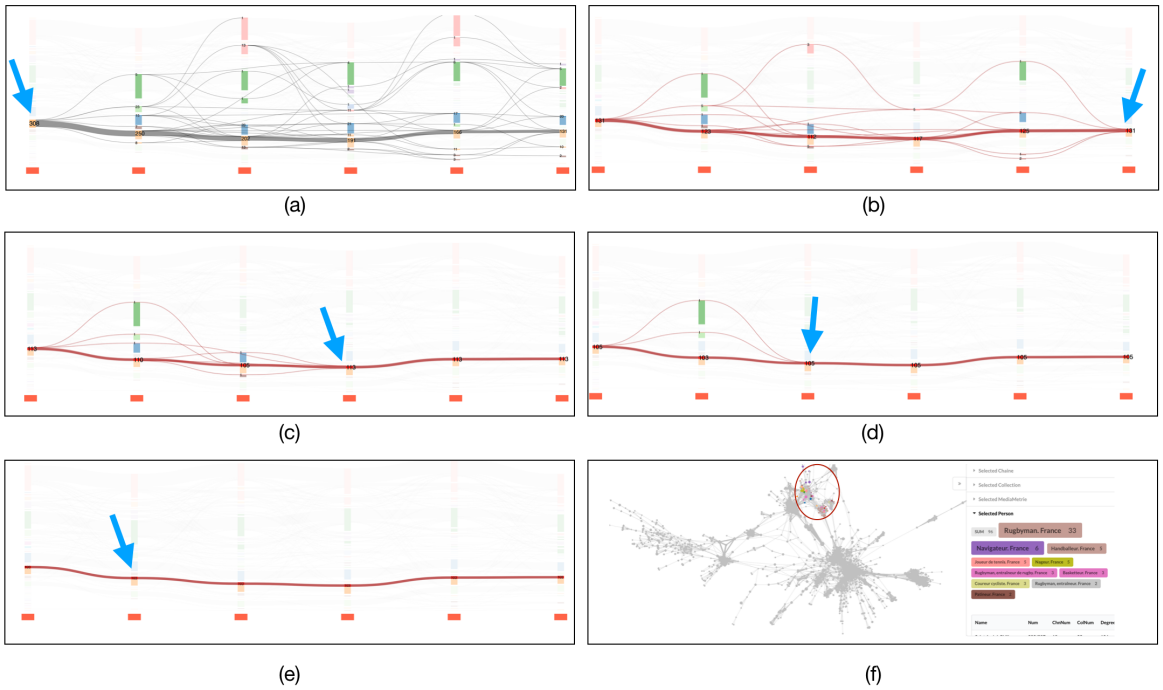
Fig. 4.29 Zoomable sankey, The user can select a time period and then continue to subdivide the graph during this time period.



Zoomable

The sankey graph is meant to make the evolution process visually clearer, but if the sankey graph has too many time intervals, it will become more visually complex. We have 6 years of data. If we use the month as the time interval, as shown in the figure, the smaller the interval division can better demonstrate the evolution of communities, but the smaller the interval will make the visualization more complicated. The zoomable feature is designed to make the visualization process easier, but users can get a deeper understanding of the smaller time interval by clicking. The specific design process is that the graphics first show the distribution of communities obtained at larger time intervals. When the user needs to know more about the community grouping in each time interval, just click on that time period, the sankey graph will be expanded and developed. The time period is divided into smaller time segments, for example, for the time period of one year, and then by the month as the time segment, for the time period of one month, and by the week as the time period.

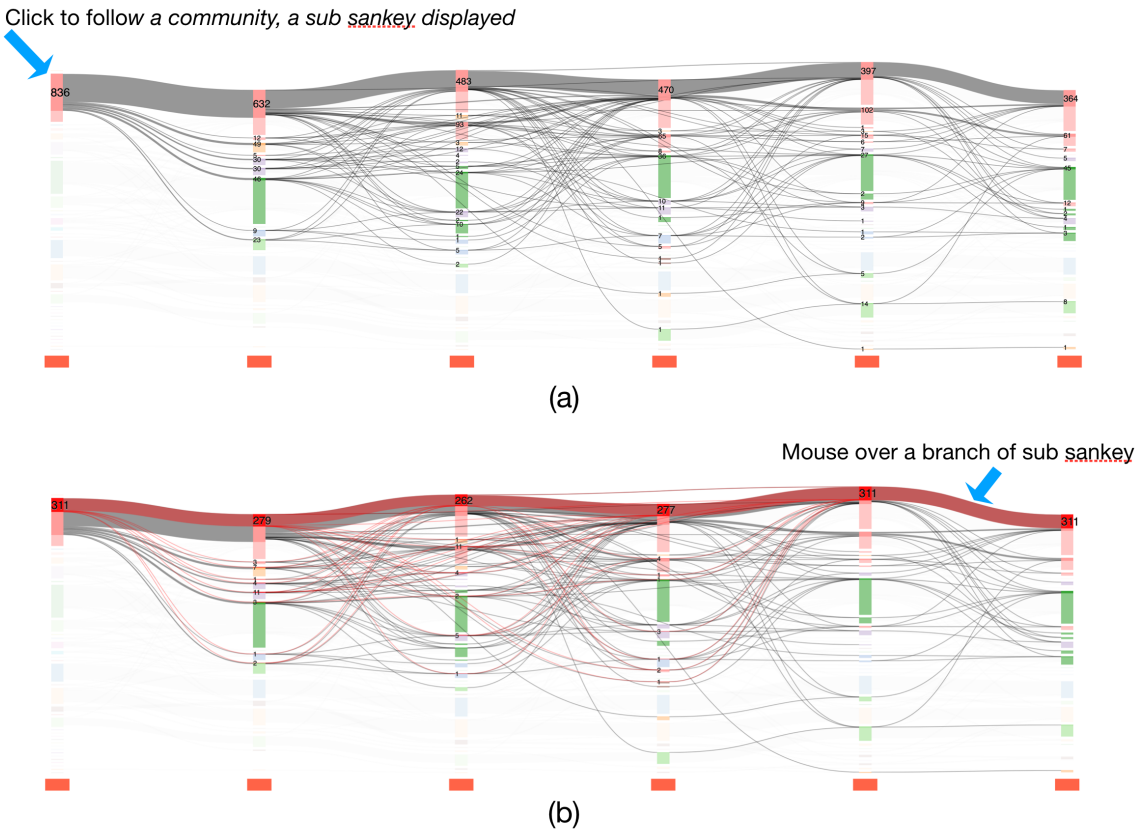
Fig. 4.30 When the user clicks on a band or a link, a subsankey consisting only of the members that are clicked is displayed, and this step can be repeated to narrow the scope.



Tracking evolution

In addition to the global view, the user can click on the connection between the rectangle or rectangle of the sankey graph to observe that in the entire sankey graph, the user can observe the evolution of the community by simply mouse hover a rectangle or a curve. Whenever the user clicks on a community, builds a sub sankey graph, the original graph is semi-transparent, and the sub sankey graph retains the original position, but the size has changed, which can be compared with the original sankey graph. The same operation can be applied to the sub ankey graph, constantly tracking more accurate member evolution.

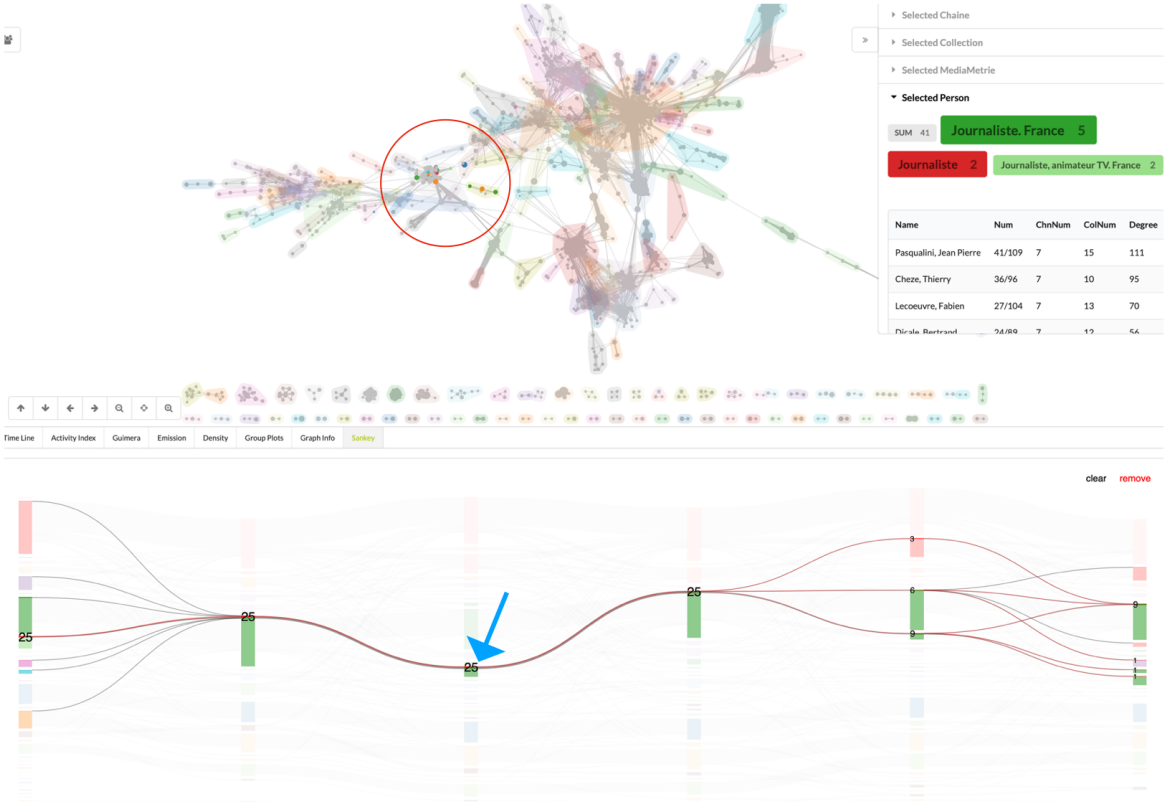
Fig. 4.31 Mouseover interaction works



Coordinate

The sankey graph, combined with the node-link graph of the global view, helps user understand the evolution of the community,the internal results of the community and the connections to other communities. User selects some node or community in the

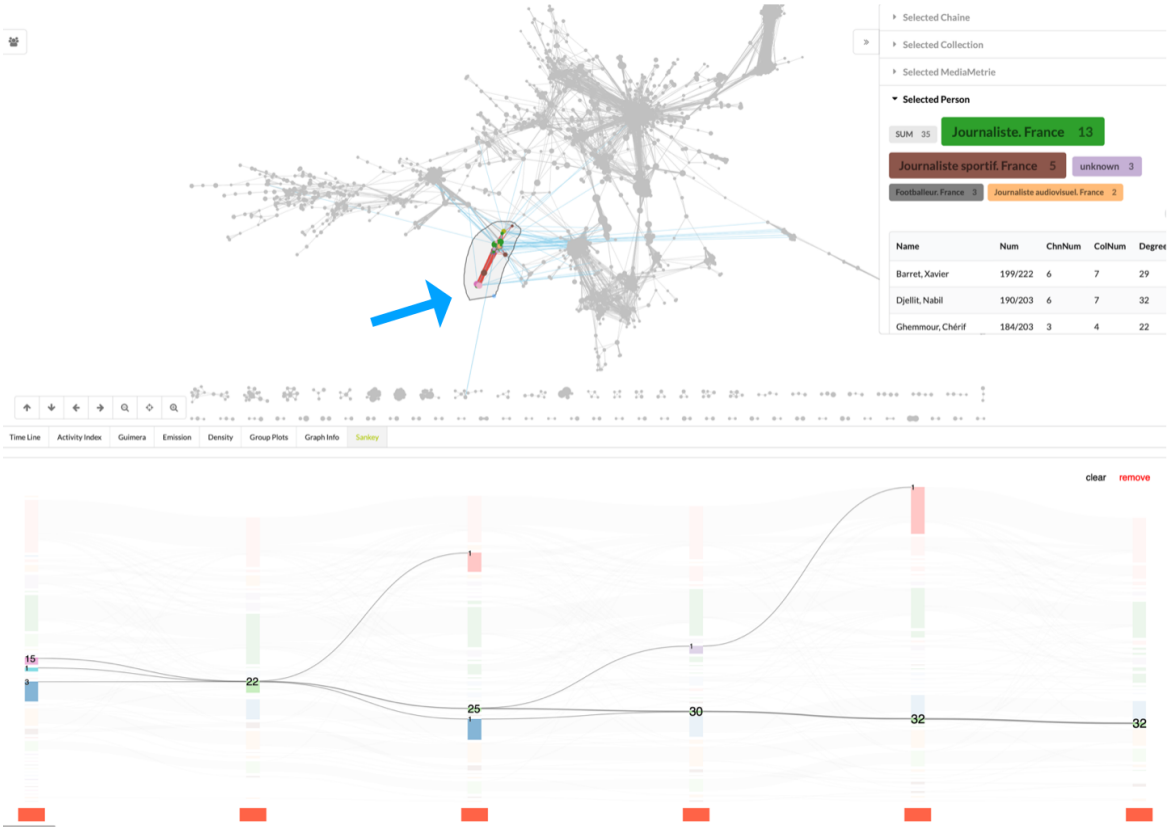
Fig. 4.32 The mouseover interaction is also available on the sankey graph.



node-link graph (based on the community obtained in all time periods), sankey graph will filter out the selected nodes in the global graph, rebuild the sub sankey, and display the sub sankey graph in the original sankey graph. Sub sankey graph shows the communities in which the selected nodes are located and their evolution over different time periods. as the picture shows. When user selects any community in the sankey graph, the user can see the location of these nodes in the global graph in the global network, as well as undersatnd the internal structure of community and its connection with other communities. This feature helps track the evolution of each node or some node.

Multiple sankeys can also be coordinated. Users can load sankey graphics for different time periods. As shown in the figure, we can use the sankey graph obtained for one year interval and the sankey obtained with three months as the time period. user can also observe the evolution of the community in another sankey by clicking on the community on any of the sankeys. This allows user to compare the differences in community between small time periods and slightly larger time periods.

Fig. 4.33 Select a community from node-link graph, the sankey graph display how the communiti's evolution.



The sankey graph is a supplement to the original dynamic node-link graph. The node-link is a spatial overlay of the dynamic graph time, and the sankey graph is a spatial extension of the dynamic time. The combination of the two is more conducive to studying the evolution of some members or groups over time.

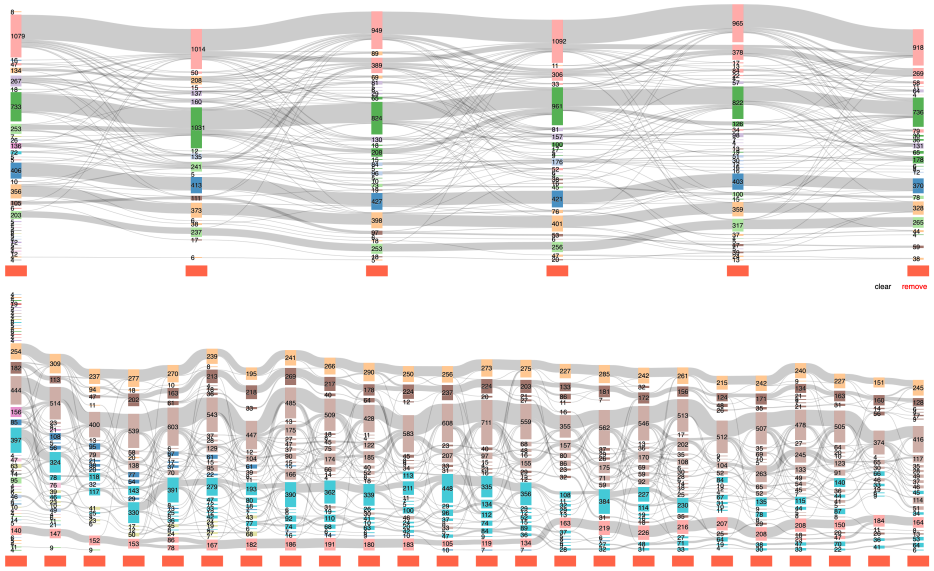
4.5 Algorithmic considerations

This section dives into algorithmic details underlying different components of the framework.

4.5.1 Comparing Simmelian backbone results

Each node in our network is a participant. For each participant, the most important attribute is the number of programs. We used non-uniform division histogram to show

Fig. 4.34 The user selects some nodes from the node-link graph, the sankey graph displays the flow of these nodes.



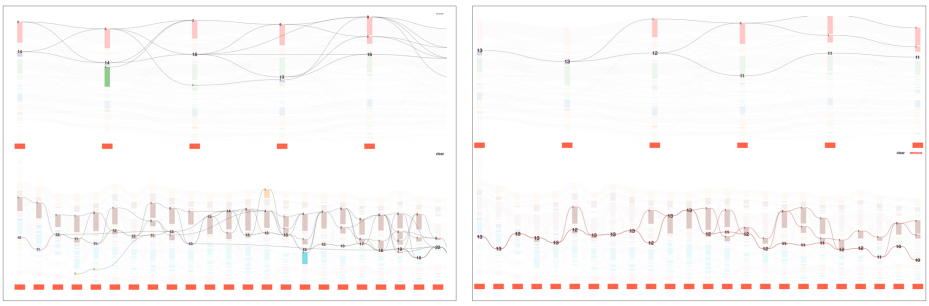
the distribution of the number of programs participants participates in. As shown in figure 4.36, the Y-axis is the number of participants, and the X-axis is the intervals of programs number in which each participant participates. It should be added that the number of programs represented by each ticker on the x-axis in the figure is not equally divided. The reason for this is that the number of participating participants in our data is not evenly distributed. The majority of participant participation in the program is relatively small. If the number of intervals for each ticker is equal, the smaller number of interval is concentrated in too many constituents. The log value is used as the y-axis coordinate to narrow the gap between different lengths..

In addition the specific practices we use for non-equal partitions are:

When we want organize data into m intervals, the interval is organized from the minimum value. Select the minimum value to the left of the interval and gradually increase the interval size. When the total number of distributions in the interval exceeds $1/m$ of all the totals, the previous value of the corresponding value at this time is used as the right side of the interval, and then this value is used as the new one. On the left side of the interval, continue with the above operation. This will result in m unequal partitions, and the number distribution of each interval is symmetric.

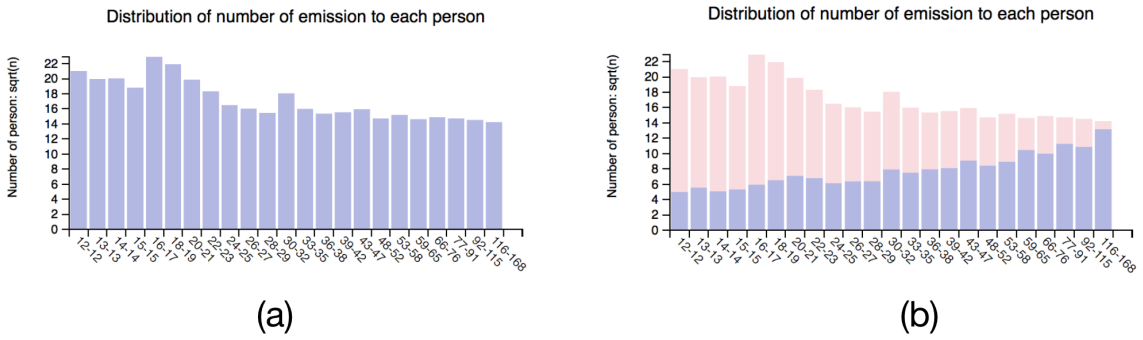
The benefits and the main use of this visualization is to visually compare the changes in the value of the network before and after the redundant data is deleted. The different colors in the figure represent the changes in the values. It can be seen intuitively that the participate frequently in the program are retained in the post-filtration graph, and

Fig. 4.35 Several sankey from different times periods can be displayed at the same time, and there is a connection between them. The user can observe the evolution of the clicked elements in another sankey by clicking on the elements in one of the sankeys.



(a) (b)

Fig. 4.36 The non-uniform division histogram shows the distribution of number of participants in different programs number interval, (a) is the attributes distribution of original graph, (b) is the distribution after using Simmelian method, the pink bars are the value filtered by Simmelian method, light blue bars are the attributes that is retained.



(a) (b)

most of the deleted participants are participating in fewer programs. Because in our research, we pay more attention to the high-visibility participants, so from the point of view of the deleted nodes, the simmelian method is effective for our data.

Similarly, we also compare the distribution of the weights of the links in the network. The weight of the link between the two nodes is represented by the number of programs that the two nodes participate in together. The more relevant participants will participate in the program, the more times they will be exposed together, the weight of their links would also be higher. We also use non-uniform division histogram to compare the changes in the weights of links before and after deletion.

As shown in the figure 4.37 the higher the weight of the links, the more retained they are in the network after deleting redundant data. From the perspective of the weight of links, it also verifies that simmelian is valid for our data.

Fig. 4.37 The non-uniform division histogram shows the distribution of number of edges in different number of programs number interval

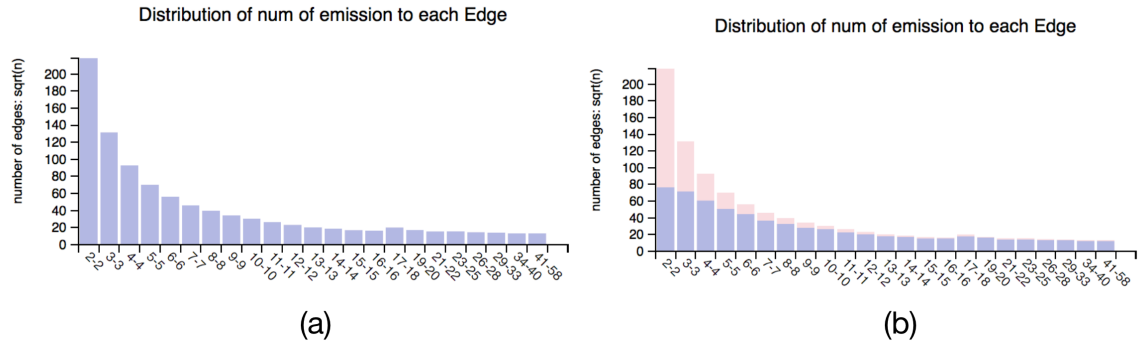
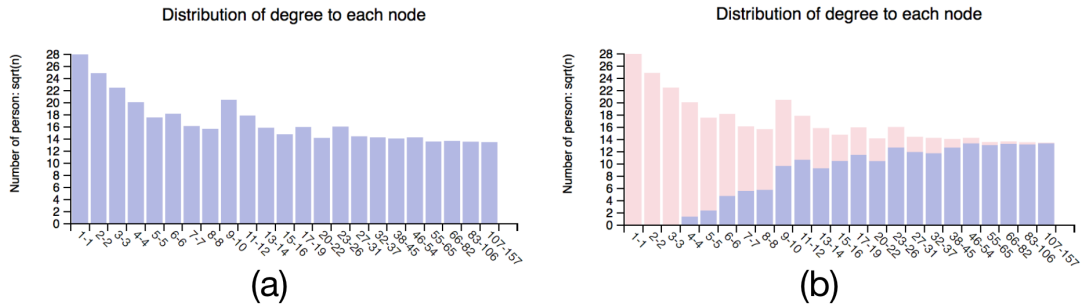


Fig. 4.38 The non-uniform division histogram shows the distribution of number of participants in different node degree interval



We also compared the changes in the distribution of nodes' degree as figure 4.38 shows. The higher the degree of a node, the more retained it is.

We use non-uniform division histogram as a research method to compare the changes of parameters before and after network change, and prove that the simmelian backbone method can effectively delete the redundant data in our data. We then explore why we choose the simmelian method and what it means to use it.

4.5.2 Dynamic graph grouping algorithms

In our work, we study dynamic community in three ways. The first way is to get the community over the entire time period, and use the line charts and scatter plots to analyze the internal activities and external connections of the community at different time periods over the entire time period. The second way is to divide the whole graph into subgraphs of different time periods. Since the subgraphs are also complex, the sb method is used to simplify the graphs. Users can observe the distribution and changes of the global community in different time periods, and can also grasp the changes and connections between communities in different time periods through drag-drop interaction. The third way is to divide the entire graph into smaller time periods, then get the community in that time period, and use sankey to visually track the change of community on the different time period.

The communities obtained in these three ways can be displayed at the same time, and in our system, they are coordinated with each other, and users can analyze and study the dynamic community from different angles.

Comparing methods and how to get multiple subgraph

In the multiple views, our subgraphs of different years are displayed at one time. The way to get the community from each submap is according to the formula x. You can also get the community according to this method, as shown in the figure.

Obviously, the community obtained by the y method is smaller than the x method, so the order of taking the method is very important. First, obtain the subgraphs in time, use the SB method to delete the redundant information, and then group the graph to get better results than using the SB method to delete redundant information, getting sub-graphs in time and then grouping the different subgraphs. The latter deletes the information on certain event segments in advance, or retains information for different time periods than the required time period. So we use the x method.

In this way, the community of the whole time period and the community within the sub-time period can also complement each other.

Comparing different method to get different sankey graphs

Because the sankey graph does not need to directly observe the node-link graph itself, and we can choose a small time period, we do not need to use the sb method to remove redundant information. The graph $g(t_i, t_j)$ of each time segment is directly grouped, and then the communities with the same members in the adjacent time segments are connected together to form our sankey graph.

In other papers, sankey is also used to obtain the global community. But this method does not apply to our sankey, because our co-participants network is too

complicated, and members of different time periods vary greatly, as shown in the figure. The sankey does not get the global community.

4.6 Use cases

We conclude this chapter by discussing several use cases showing how the framework indeed supports the analysis of the OTMedia data.

4.6.1 General idea about French TV/Radio participant network

Programs type	programs Number
Magazine	169524
Journal televise	141017
Interview entretien	63504
Tranche horaire	30885
Debat	10340

There are many different types of programs for 6 years. So we have made different graphs according to the program category. A comparison of several different graph before and after the SB method is shown, such as magazine, debate(figure 4.39), news(figure4.40), talkshow(figure 4.41) etc. Different graph have different characteristics, and different results can be obtained by using the SB method. For example, in the news program guests as figure 4.40 shows, it is difficult to find communities, because the participation of the guests is relatively random.

Figure 4.42 shows the graph of the magazine programs. It can be seen from the figure that the community composed of the guests has a clear correlation with the profession. Politics, sports economy and culture are also the most important themes. Journalists are interspersed between different communities.

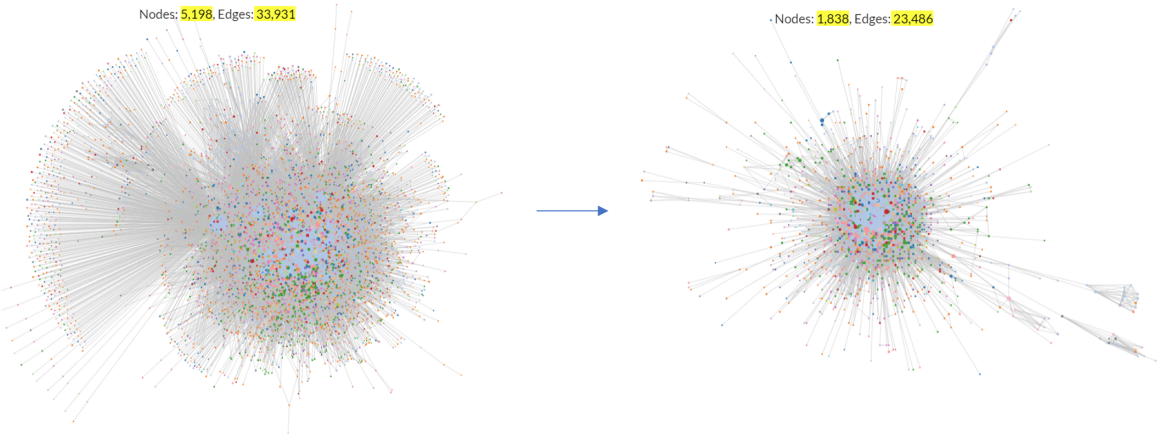
Global graph can present an overview of complex data, and our system enables users to drill down and explore data from different layer. The following use case shows the usage of analysing sport communities.

4.6.2 Sport and football

From the magazine network, ing selecting the sport communities, we can see that the guests participating in the sport accounted for a large proportion of the number of

Fig. 4.39 Network built from debat programs, left is the original graph, right is the filtered graph build from using Semmilian backbone method.

Fig. 4.40 Network built from news program, it is difficult to find communities from it.



people in the French magazine type program. As shown in the figure 4.43, in addition to internal contacts, sport guests also have a lot of connection with journalists and politicians. Among the sports guests, the largest proportion comes from football followed by rugby, and then sports-related reporters. Figure 4.44 shows the layer of sport which could be extracted by our framework.

It can also be seen from various indexes (figure 4.45) that the sports community is quite active at all times and the activity in summer is higher than in other occupations.

The financial windfall of football If we look more closely at the figure relating to the "Group Plot" (figure 4.45)for football:

Fig. 4.41 Network built from talkshow.

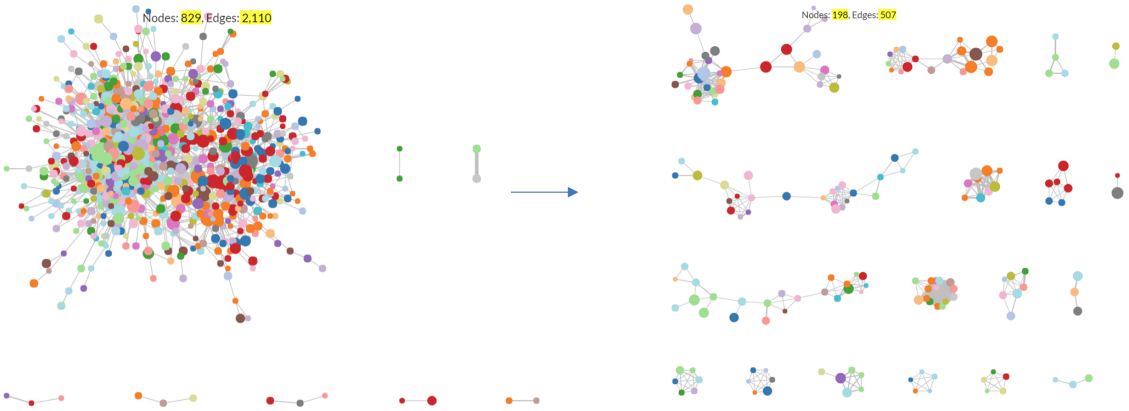
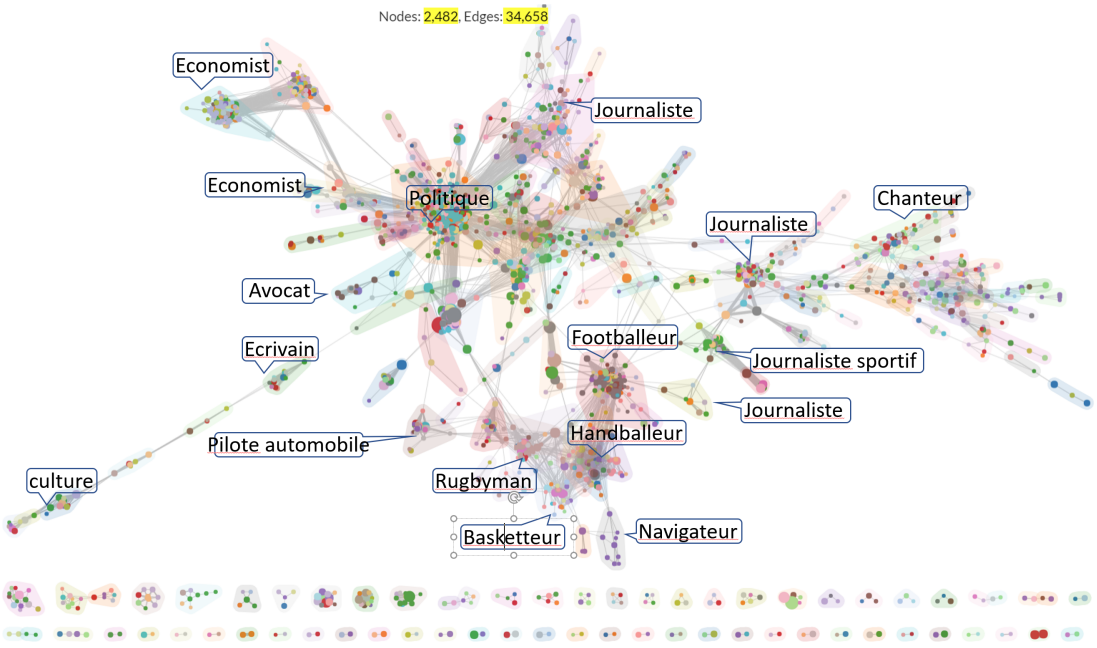


Fig. 4.42 The overview graph built from 'magazine' TV programs



2 types of competitions with different distribution methods: football rights are bought private broadcasting with very expensive rights, these are the competitions "by clubs" (French league, French cup and League Cup, + European Cup of clubs) and by countries worldwide, the euro cup and the world cup. There is a legal obligation to broadcast on a non-paying channel as soon as there is the French team in play and we can observe all this on the diagram, big events are not broadcasted by Canal+. TF1

Fig. 4.43 The connection between sport communities and other communities

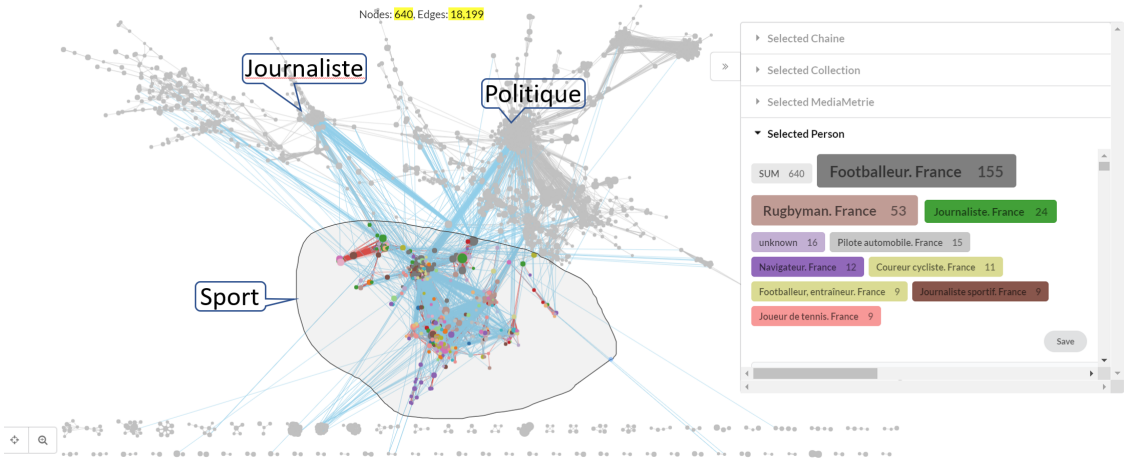
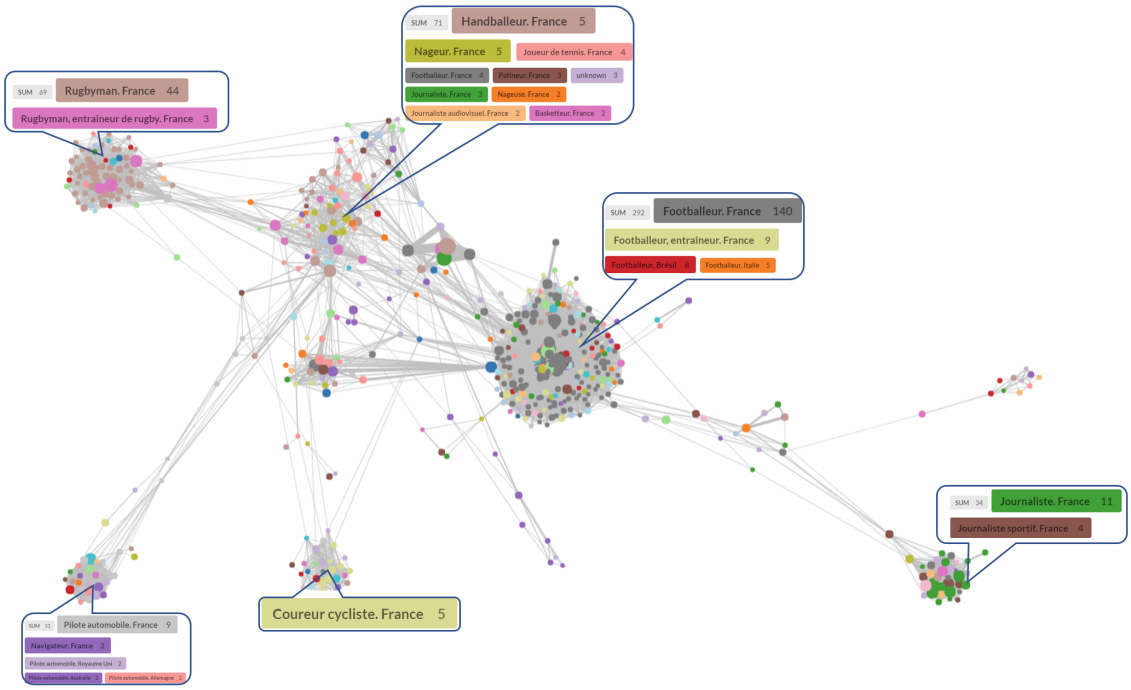


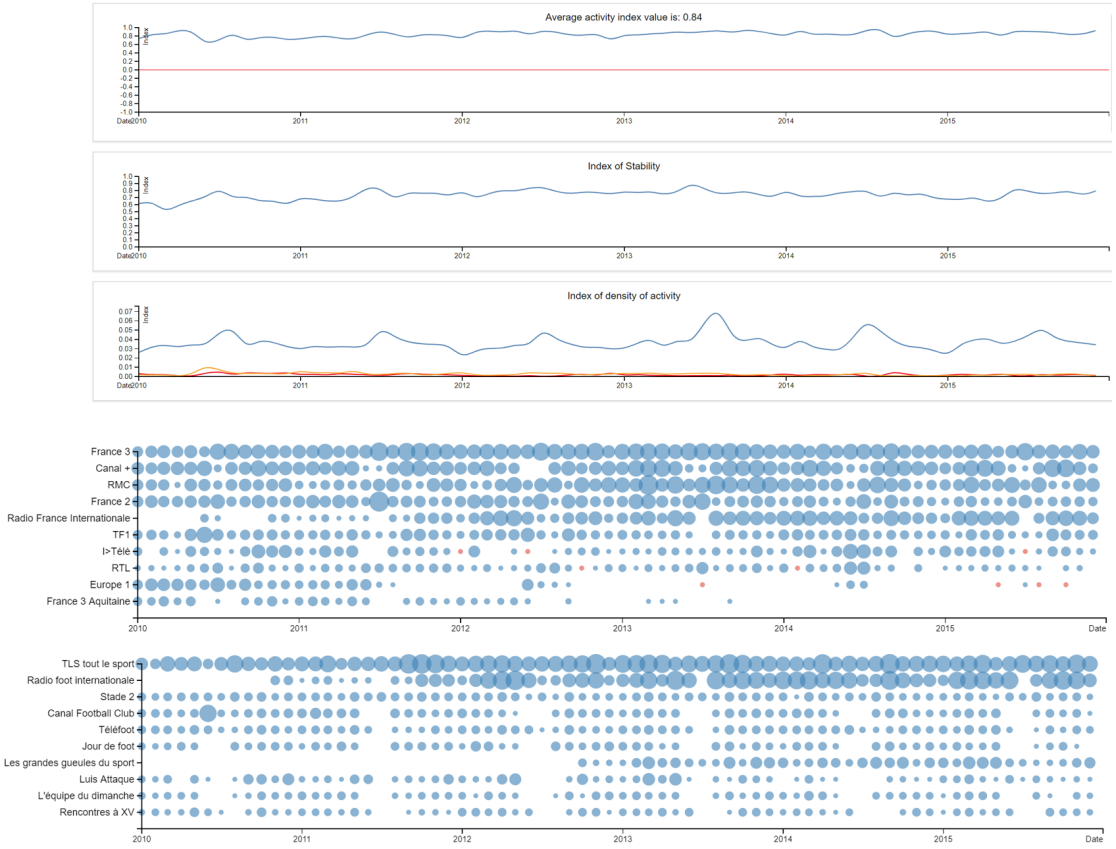
Fig. 4.44 The composition of sport communities



acquires rights to have audience, otherwise France 3 or 2 it depends on the budget of the year!

Are footballers present elsewhere? Combine the node-link graph with the sankey graph. If we select someone from node link graph, we can observe in the

Fig. 4.45 Indexes of all sport community

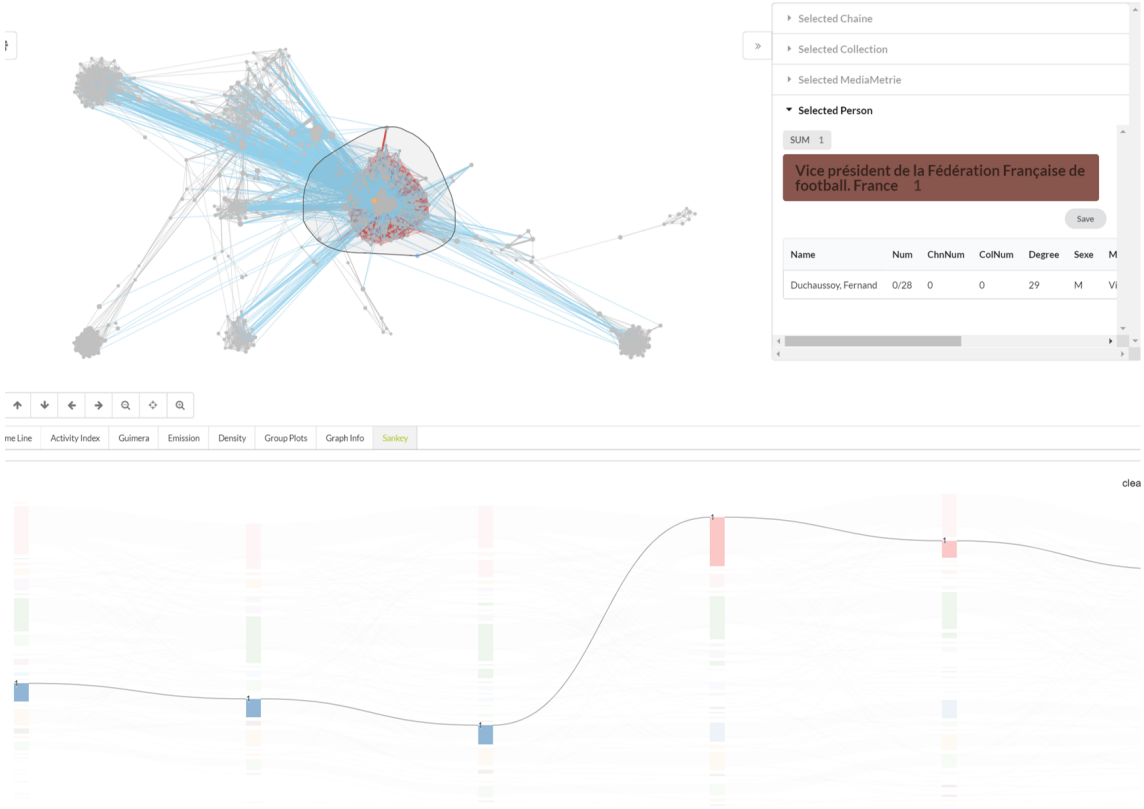


sankey graph that the person's community changes over time. As figure 4.46 shows, the selected person 'Fernand Duchaussoy', we could observe that in the first three years, he is in the community of sport, but the following three years he changes to the community to politics. This is caused by changes in his profession.

The composition of the community will also be related to the radio and the program: Contact of the program and station that the same community participates in. Select a community to see all the TV stations and programs that it participates in. We will observe that some communities are together because of a TV show, and some communities are participating in different programs on multiple TV stations.

Structural changes in the same community member in different stations In order to compare and observe the changes in the structure of the same community in different TV stations and different layers, we can display the graphs of different layers through our framework. Through the drag-drop function, it is possible to observe the specific change process of the same community in different layers.

Fig. 4.46 Example of someone in different community at different times.

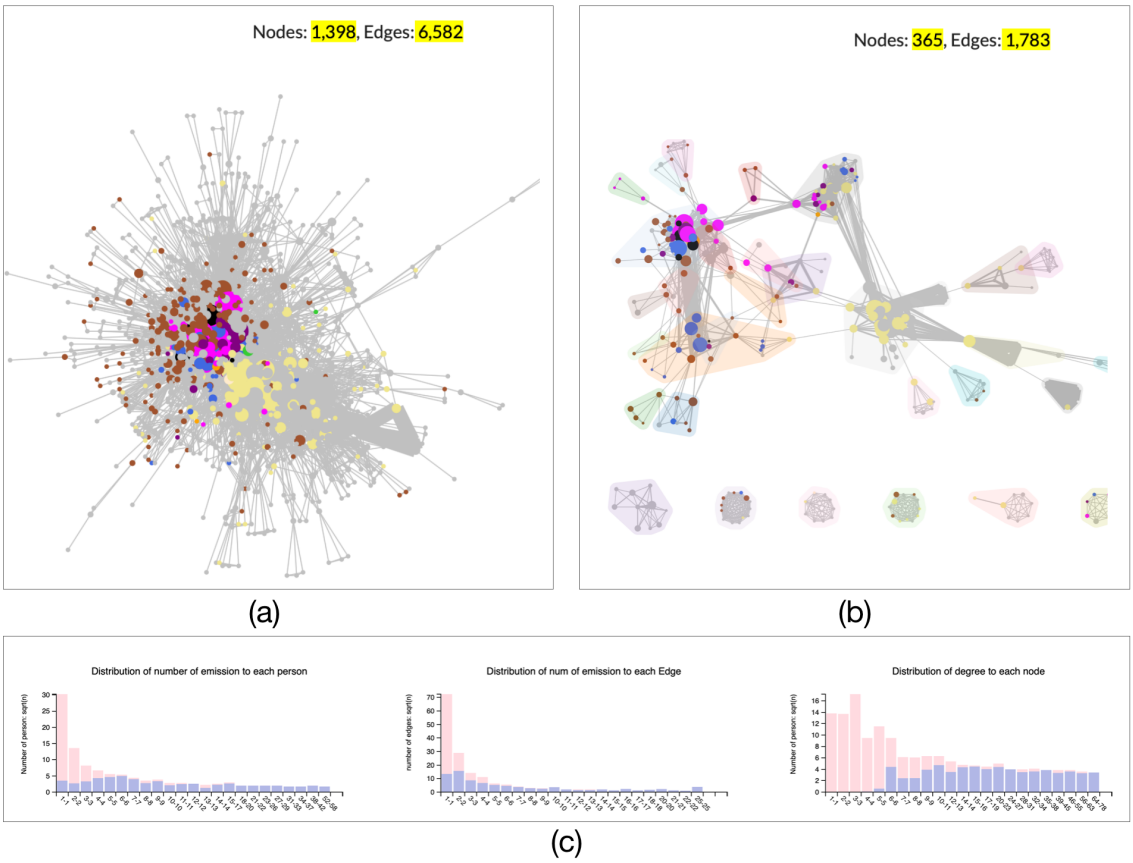


4.6.3 Party members in TV programs during 2017 France election

During the 2015 French elections, we formed a network of politicians participating in the program. The initial graph is very complex, we use the SB method to simplify it. As we can see from the histograms (figure 4.47 (c)), it keeps the important nodes and links. The relationship between the community and nodes in the picture is immediately clear. The color of the nodes is based on the party in which they are located.

The diagram helped ina researchers analyze the relationship of politicians participating in programs on television and radio during the election and observe the connections between parties in television programming.

Fig. 4.47 During the 2015 French election, different political parties participated in the network of programs.



Chapter 5

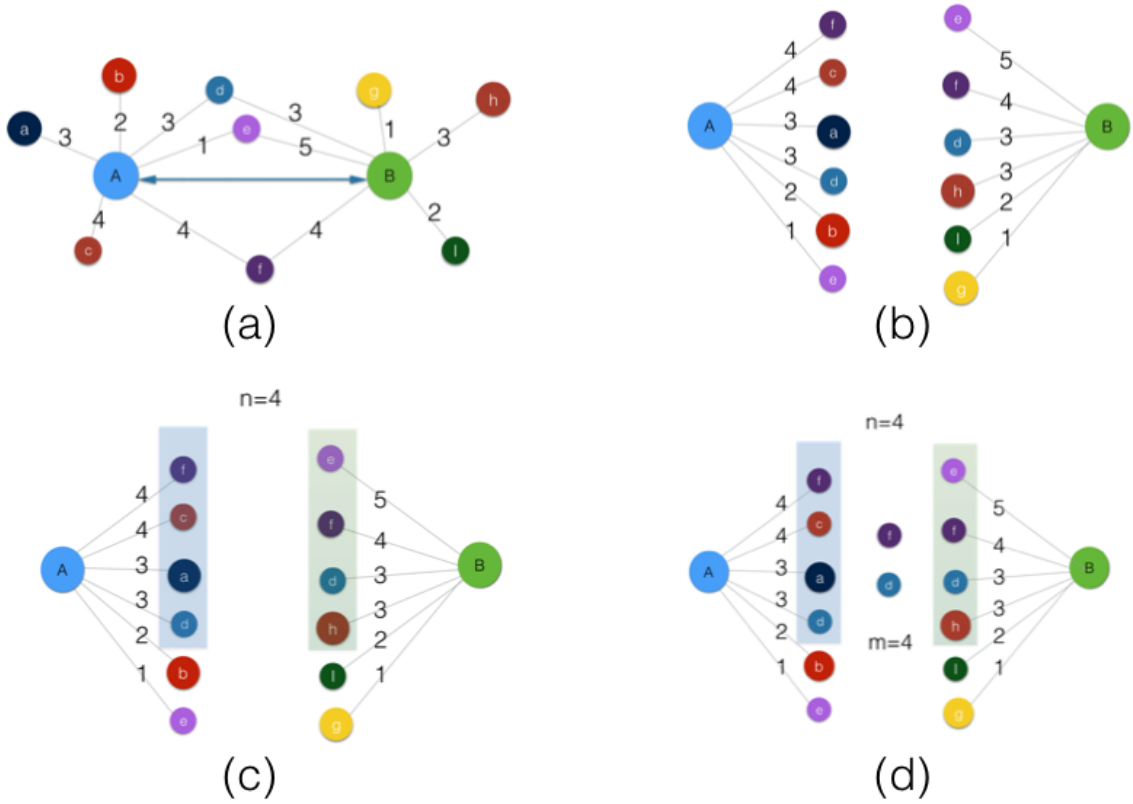
Reducing link complexity with the Simmelian backbone

This chapter focuses on algorithmic aspects surrounding the use of the Simmelian backbone [48] to simplify the graph structure. While the display of the backbone aims at simplifying the graph representation offered to the user, its computation requires the setup of sensitive parameters. In order to be confident that the Simmelian backbone was a relevant choice for our framework, we embarked into a journey to better understand the impact of these parameters on the output provided by the algorithm. It is this algorithmic quest we report on.

The backbone is computed based on two metrics: *edge strength* and *edge redundancy*. The strength of edges, (globally) denoted as m, can either be inferred from given attributes on the data or computed from the graph structure. Edge redundancy, denoted as n, depends on m. After edge strength and edge redundancy are computed, a backbone is obtained by filtering out edges with low redundancy.

In their original article, the authors do not provide a thorough discussion on how strength and redundancy behave or how they can be used to somehow steer the backbone. The article makes a clear case about the backbone providing a meaningful simplification of the graph structure. We thus aim at bringing light on how parameters impact the obtained backbone, focusing on our dataset.

Fig. 5.1 Process of Simmelian backbone



5.1 Edge strength and edge redundancy

Edge strength, as the name suggest, is intended to reflect the relative importance of an edge in the network. In our case, edges support TV programs to which invitees (incident nodes) have participated. Thus, the number of such programs comes handy and naturally defines edge strength: an edge is as important as the number of programs it supports.

Once edge strength has been defined, edges incident to a node can be sorted in decreasing order (listing more important edges first). Edges of equal importance do not need to be sorted among themselves as we shall see. As a consequence, neighbors of a node u can be ordered just the same (neighbors are as important as the edge linking them to u). We will refer to this order when talking about the strongest neighbors of a node u.

As figure 5.1 shows, given a node $u \in V$, denoted by $E(u) = [e_1, e_2, \ldots, e_p]$ the ordered list of incident edges to v. All edges $e_1 = \{u, w_1\}, e_2 = \{u, w_2\}, \ldots$ have u as one of their incident node where nodes $w_1, w_2, \ldots$ are neighbors of u. Depending on the context, $E(u)$ will denote either the set of ordered edges $[e_1, e_2, \ldots, e_p]$ or the set of ordered neighbor nodes $[w_1, w_2, \ldots]$.

Let $e = \{u, v\}$. Given an integer m, denoted by $E_m(u)$ and $E_m(v)$ the m strongest neighbors of u and v. Edge redundancy, denoted as $\rho_m(e)$, is the number of common neighbors among those, that is $\rho_m(e) = |E_m(u) \cap E_m(v)|$. (Note that $E_m(u)$ may contain more than m nodes when there are multiples nodes of equal strength.)

The second parameter, n, used to compute the Simmelian backbone is a threshold to filter out edges of lower redundancy. The Simmelian backbone of a graph G, denoted as $\mathcal{S}_{m,n}(G)$ is thus obtained by filtering our edges $e \in E$ such that $\rho_m(e) < n$.

5.2 Studying the behavior induced by parameters m and n

Clearly, both parameters only cover a finite range. Valid values for m and n all lie within $[1, \max_{v \in V} \deg_G(v)$ with a much lower maximum value for n.

We calculated all n, m in the range of $1 - 30$ by enumeration as variables, and compared the results of different parameters. According to the definition of Simmelian, the value of n represents the top n neighbors to which each node is most closely connected, the value of m represents the same number of top n neighbors of two connected nodes. In our data, the basis for judging whether it is closely connected is the weight of the link between the node and its neighbor, that is, the number of programs that participate together.

We designed a two-dimensional plot to observe the results. For the original network, in the Simmelian method, when we use different m, n to get the new different results of the deleted redundant information network, we compare the new obtained graph with the original graph to find its ratio of changing. The network to which each group (m, n) arrives is unique, and the number of nodes and links of each network is also fixed. We use the number of nodes and links in each network obtained after the Simmelian backbone to take the percentage of the original nodes and links as the abscissa and ordinate of the plot axis, each different (m, n) in the plot correspond to a point, for example, the original network has 100 nodes, 200 links, while we use m=4, n=10, the new network after deleting redundant data has 40 nodes, 100 links, then in our plot the corresponding point has an abscissa of (0.4, 0.5). The reason we take the percentage

is that it is not limited to the number of original graphics nodes and links. We only look at the percentage after deleting redundant data to compare the effects of different original graphics. We enumerate all possible $m, n(m < 30, n < 30)$, and show the points corresponding to the parameters in the plot for comparative analysis.

5.3 Different types of value to quantify Simmelian relationship

We designed a two-dimensional plot to observe the results. For the original network, in the Simmelian method, when we use different m, n to get the new different results of the deleted redundant information network, we compare the new obtained graph with the original graph to find its ratio of changing. The network to which each group (m, n) arrives is unique, and the number of nodes and links of each network is also fixed. We use the number of nodes and links in each network obtained after the Simmelian backbone to take the percentage of the original nodes and links as the abscissa and ordinate of the plot axis, each different (m, n) in the plot correspond to a point, for example, the original network has 100 nodes, 200 links, while we use m=4, n=10, the new network after deleting redundant data has 40 nodes, 100 links, then in our plot The corresponding point has an abscissa of (0.4, 0.5). The reason we take the percentage is that it is not limited to the number of original graphics nodes and links. We only look at the percentage after deleting redundant data to compare the effects of different original graphics. We enumerate all possible $m, n(m < 30, n < 30)$, and show the points corresponding to the parameters in the plot for comparative analysis.

In the original paper, number of triads are used to sort the neighbors of each node. It is also mentioned in the text that other weights can be used for sorting. The sb method determines whether the relationship between two nodes is tight, depending on whether the two nodes satisfy a certain number of important common neighbors, and the importance between each node and its neighbors can be obtained by different methods. The method described in the original text is to rank the importance of the node and its neighbors according to the number of triads (the number of identical neighbors). In our data we use the weight (number of programs) between nodes to determine the importance between nodes and neighbors.

For two nodes: x, y are connected by edge x, y, we use nb(x) to represent all nodes connected to x, and nb(y) means all nodes connected to y. The sb method is: First, nb(x) and nb(y) are sorted in reverse order according to a certain method to obtain list (nb(x)) and list(nb(y)). Then the first n elements of list(nb(x)) and list(nb(y))

are represented as $top_n(list(nb(x))top_n(list(nb(y)))$. If there are at least m identical elements in top1 and top2, then the relationship between nodes x and y is valid. If m(top1, top2) is less than m then the relationship between nodes x and y is redundant.

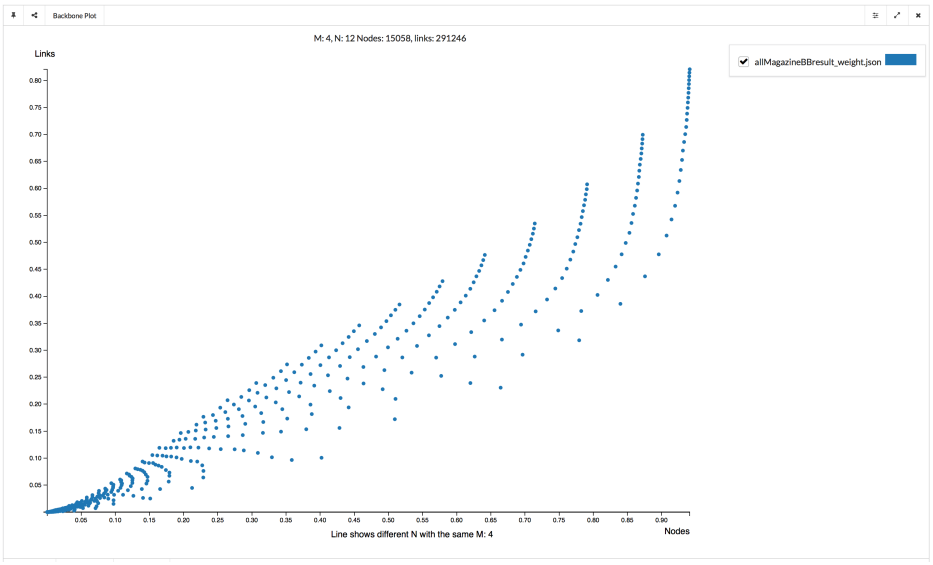
The method of ordering nb(x) and nb(y) is open. If sorted by number of triad, it is the order of the number of neighbors between each node connected to x. In our graph, one of the important criteria for determining the relationship between two nodes is the number of programs that two nodes participate in together. The weight of the graph is also the number of programs. So we use weight to sort list(nb(x)) and list(nb(y)).

We compared the results of number of triad and weight sorting. The same (m, n) is chosen and the results obtained using the sb method under two different conditions.

The comparison shows that the number of triad retains more links with lower weight values, while the reserves of nodes do not change significantly. So we use weight as the basis for sorting. Similarly, we also compare the use of other different characteristics as a sorting basis: such as number of channel or number of collections, as shown, the results are not as effective as programs. It may be more efficient to combine the same features as a basis for sorting, but for our co-participant network it is already effective to use weight as a sorting basis. If we need a more precise study, we can optimize it more deeply in the future.

5.4 Different parameters of the same network

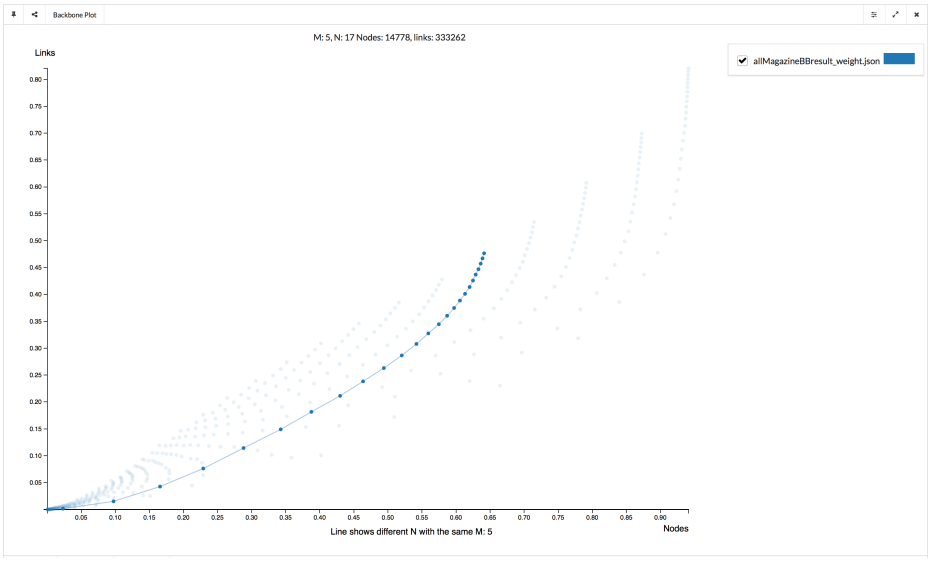
Fig. 5.2 The plot of different results from different Simmelian backbone parameters. Each node correspond to different parameter pairs of m and n



As shown figure 5.2, each point represents the distribution of the percentage of the number of nodes and links for each different network (m, n) obtained from the same network. Although the points in the figure have some rules, they are still messy. But when we connect the points with the same m value (as shown in Figure 5.5) or the same value of n (as shown in Figure 5.6), the pattern of the effects that changes in m and n have on the filtered results becomes very obvious.

- Take the same M, observe with n as a variable. As shown in figure 5.3, we take the points where M equals 5 to connect together. We will find that as N grows larger, the less redundant data is deleted by the Simmelian backbone, that is, the filtered network retains more nodes and links. Moreover, the retention ratio of nodes and links is different. The ratio of nodes reserved at any point is higher than that of links. This is because each node has multiple links connected, and links are more redundant.

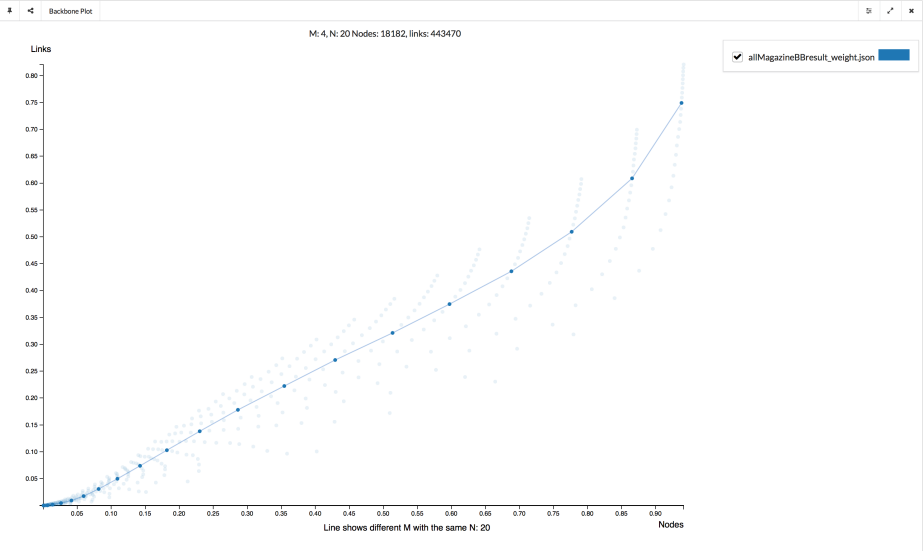
Fig. 5.3 Dots with same m = 5 are connected in turn



It is worth noting that as the value of N is larger, the proportion of links and nodes that are filtered out is slower. Because as n increases, when n is greater than the degree of nodes, the increase in n does not affect the decision of redundancy. So the slower the rate at which links and nodes are filtered out, when n is greater than the maximum of all nodes' degrees, the change in n no longer affects the ratio of nodes and links reserved. As shown in Fig 5.3, the point where M is equal to 5, when N is greater than 20, the effect of the proportion of n changes in nodes is gradually reduced. The reason why the percentage of nodes retained

is less than 75% is that the degree of 25% of nodes is less than the judgment condition $M = 5$, that is, 25% of deleted nodes are those whose degree is less than 5, regardless of the influence of N.

Fig. 5.4 Dots with same n = 20 are connected in turn



- As shown in Figure 5.4, we connect all the points of $N = 20$ in turn to observe the change law caused by M value. In the plot, the smaller the M value, the more nodes and links are retained, and in terms of the rate of change, when the value of M changes, the percentage of nodes retained is slightly higher than the retention ratio of links, but the change in value of N causes the change ratio of links to be much higher than the proportion of changes in nodes. The smaller the M value, the higher the rate of change that links and nodes retain.

- Connect all points with the same value of N. It can be observed that each line corresponds to a different M, and the variation law caused by N is basically similar. The difference between two adjacent N value curves is small.

- Connect all points with the same M value. Each line corresponds to a different M value, and the difference between the two adjacent M value curves is larger.

By comparing the effects of M and N on the network results, we can find out that:

- The change in the M value has a more significant effect on the result than the N value.

Fig. 5.5 All dots with same M value are connected in turn

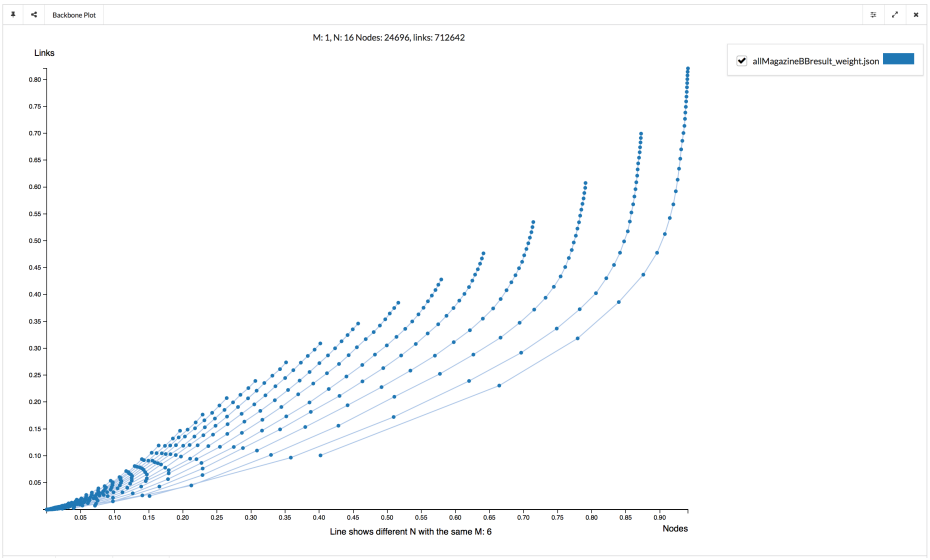
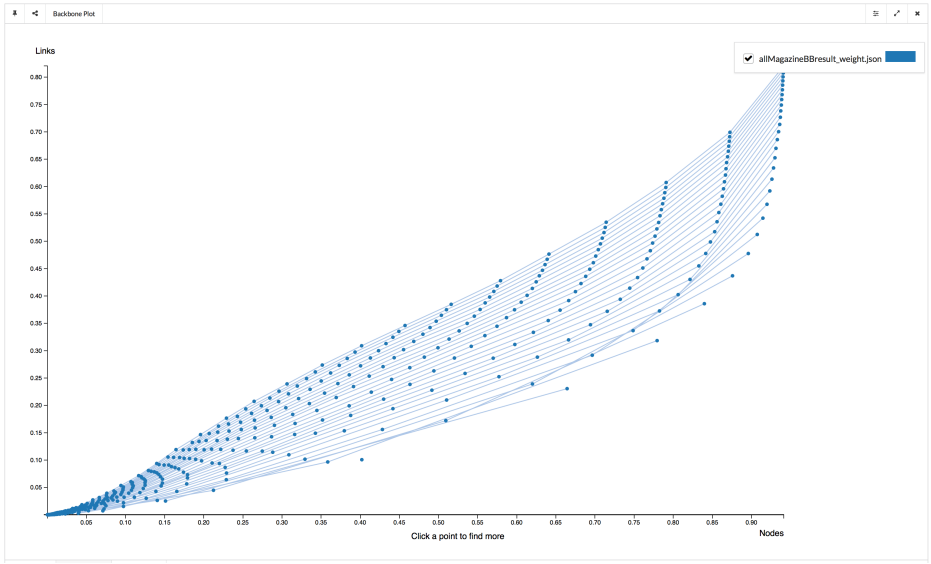


Fig. 5.6 All dots with same N value are connected in turn



- When the M value is too small or the N value is too large, the algorithm has no effect on the screening of redundant information. How to choose the right m, n is the key to using the Simmelian algorithm. We tested the m and n variable curves of different graphs. It can be seen that the different network, m, and n variables have similar effects on the results, but the values of m and n are not well based on the standard. Currently m, n choice is based on the results, to the

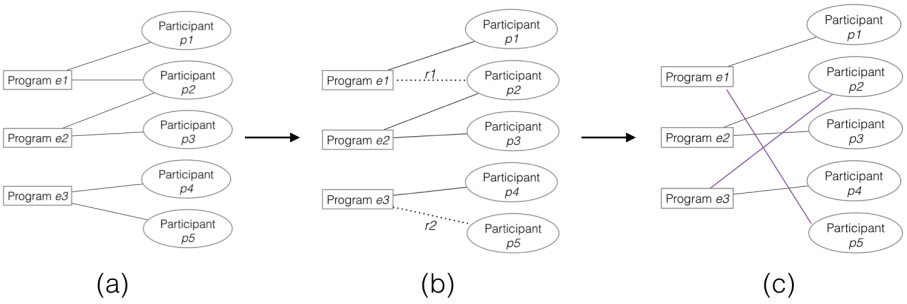
artificial judgment and to choose the appropriate m, n values. The choice of m, n values are also very different in different graphs.

5.5 Is the Simmelian method beneficial for the detection of communities

An important condition for detection of communities is that the community internal relations are closer than the external ones. In order to explore whether the Simmelian method is beneficial to find communities, we have designed a method to apply the Simmelian method to random graphs and study the influence of m and n parameters. The random graph here is random to the original data. Taking the co-participant network as an example, the reason for the existence of communities is that many participants participate together many times, that is, they appear in the program together more frequently than other participants, community internal members interact more frequently, and they often appear in the same program. On the contrary, if there is no community, the guests appearing in the program are random, and there is no correlation between them.

Random participating in the program

Fig. 5.7 A random process of participating guests in the program.

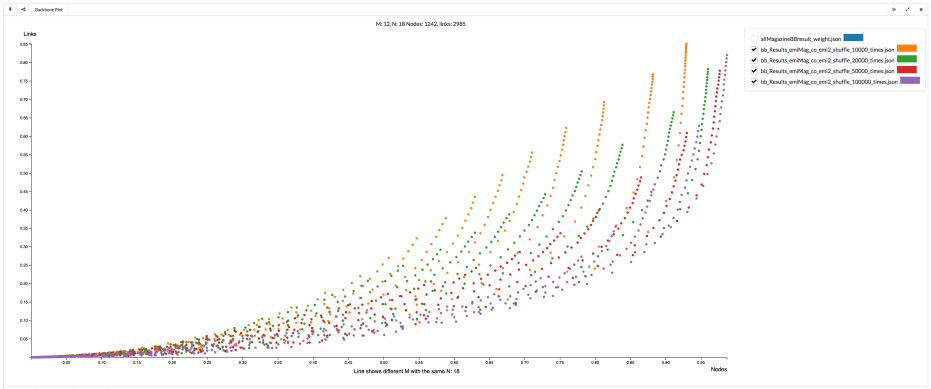


In the original data, we randomly select two programs, and randomly select two participants from those two programs to change their positions. Then we repeat this random exchange process a certain number of times. As the number of exchanges increases, the randomness of the guests in each program will be higher, and the larger the difference between the network and the original network will be.

The specific approach: Two program e_1, e_2 are randomly selected from set of programs E. Randomly select one participating guest P_1 from program e_1, Then

randomly select a guest p_2 from the participants of program e_2, then exchange p_1, p_2 position, after exchange, p_2 guests participated in program e_1, p_1 guests participated in program e_2. Repeat the above operation a certain number of times, so as to not destroy the number of invited guests of each program, just assume that each program is a random invited guest. We then establish this randomly generated social interaction relationship as a network, and then use the Simmelian backbone method with different parameters to judge the results obtained.

Fig. 5.8 Compare the results of the network in Simmelian after different random processes



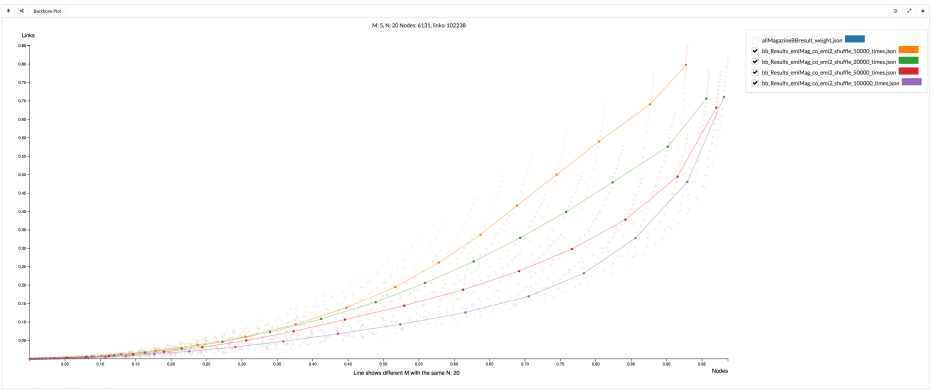
The figures show the network obtained after a random exchange of the original data, we use different (m, n) parameters of the Simmelian method, the x-axis and y-axis corresponds to percentage of the number of nodes and links in the new network to that of the original network. The orange dots are the result of random exchanges of 10,000 times, the green dots are the result of random exchanges of 20,000 times, the red dots are the result of random exchanges of 50,000 times, and the purple dots are the result of random exchange of 100,000 times.

It is intuitive to observe that the random number increases and the result is shifted to the right. As shown in the figure, we connect the points of the same M value together. Figure 5 is to connect all the points with the same N value together. As the number of random times increases, the curves move to the right, which means that the retention ratio of links is decreasing, but the proportion of nodes is more complicated.

Similarly, we have studied the characteristics and differences of network by using the M and N curves respectively.

When we use N=20, the points corresponding to different M are connected together, and the network obtained by different random exchange times corresponds to different curves. The curve is clearly shifted to the right, and the slope rate is gradually

Fig. 5.9 Connect all the dots with same N value (n=20), different color of lines stands for different random processes result



increased as M decreases. The right curve of the curve means that the retention ratio of links is reduced, that is to say, the network after randomization becomes loose, and the connection between communities is reduced, which means that the existence of community is weakened, and when M is small (M<6) that is obvious. To see, the same (m, n) although the retention ratio of links decreases with the increase of random times, but the retention ratio of nodes increases with the number of random times, because of the random social process, each probability that individuals will participate in the program will increase, and the probability of each person's contact with each other will increase. Therefore, the probability of having a common understanding between two people will increase, and the retention ratio of nodes will be high, but with the increase of m (m>=6), the retention ratio of links and nodes of the same (m, n) graph decreases as the number of random times increases. This shows a random increase, and the chances of having more important friends between the two (the data in the figure are six) are reduced.

Fig. 5.10 Connect all the dots with same M value (m=5)

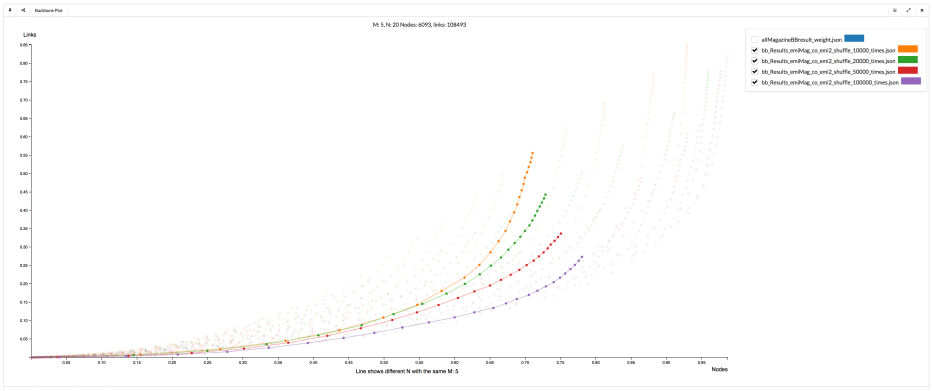


Fig. 5.11 Connect all the dots with same N value

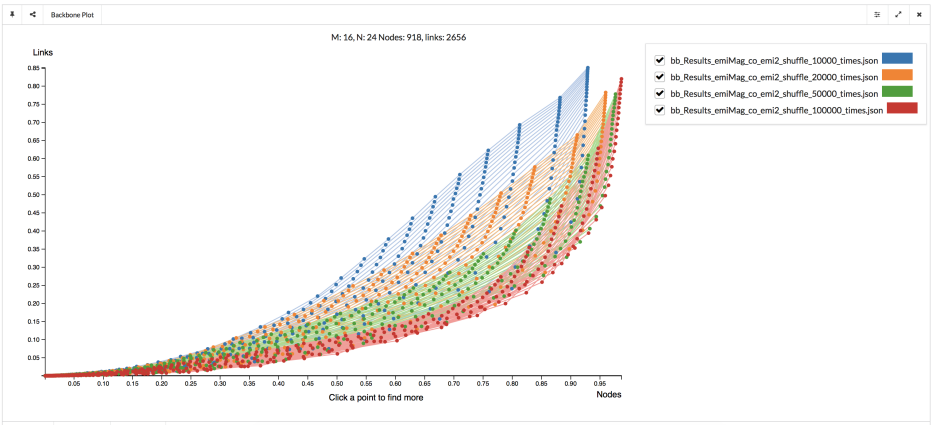
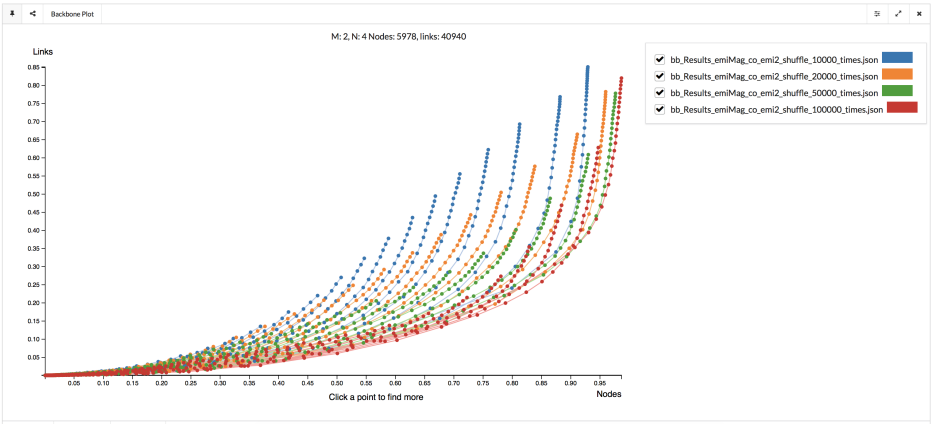


Fig. 5.12 Connect all the dots with same M value



When we use M=5, the points corresponding to different Ns are connected together. As shown in the figure, the trend and the M-value corresponding curve are roughly the same.

From the comparison with random graphs, we find that the network in which communities exist has a higher ratio of links after using Simmelian than the network with random relationship, because the connections within communities are more tight; using the Simmelian method in the parameter M value, larger nodes have a higher percentage of saves because there are more common stakeholders among each member of the community. It can be seen from this that the network obtained by the Simmelian method can well preserve the characteristics of the community and help to study the communities in the complex network.

Tests on Simmelian results can be further studied. The size of points in the graph can be used to show the number of communities obtained by different results. And this

test method can be used to verify whether there are communal features in different networks.

Chapter 6

Conclusion

In this thesis we focus on data modeling, visualization and analysis approaches toward a better understanding of complex network and time evolving network. The main challenges we addressed are how to effectively index and understand the information in massive video documents by using visualization, and how to discover the existence and evolution of communities in dynamic networks, and how to use visualizations more interactively to analyse the evolution of a community. The paper emphasized the importance and consequences of interactive visualization in analysing multiplex and dynamic networks. Two visual frameworks were designed and built for solving those challenges.

- For the video documents labeled by texts, we extract the concepts and model them into an interactive network. We proposed our own hierarchical clustering method to form groups, and create the visualization system called "*Visual Cloud*" to interactively display and analyse the multiple concepts. The innovation of this visualization system is to display the character image and text into a hybrid tag cloud, augmented with a heatmap to highlight the key points.

- For dynamic graphs, the paper studies and improves the Simmelian backbone method to remove redundant data to offer users a simplified network structure. The analysis of dynamic graphs is tackled by combining different aspects and interaction into a unified visualization framework. The system combines node-link graph with Sankey diagrams to reveal the evolution of communities. The system offers a variety of coordinated visual components, users have many possibilities to use the tool, and to explore the data with some new effective interaction methods such as drag-drop feature.

The design of our multimedia system completely falls into the search task as described by Brehmer et al.'s typology consisting of two parameters: search location and search target, either being known or unknown. Each situation maps to a subtasks: lookup, locate, browse and explore. Usual search engines are often used for lookup tasks (with location and target both known). Video broadcasters such as Youtube link videos together to support browsing tasks (when the location is known but not the target). Locating tasks consists in knowing the target but not the location, made successful by keyword search.

The visual cloud supports exploration and browsing tasks, our keyword search is too strict to provide proper lookup task support. It should be improved, together with video linking, to better support lookup and locate (beyond time location with the timeline). One last important future work concerns comparison tasks. We currently refine information through visual cloud hovering, timeline browsing, and leapfrogging. However, beyond side-by-side comparison of two queries tabs, we do not have explored other means of comparison. This need quickly rises as we would like to compare periods of time.

Laputa focus on interactions that seem relevant to manipulate and navigate multiplex networks and dynamic communities. Currently it just works for our media data, the future work direction will be to satisfy the application of the system to a wider range of data. We have created an interface that converts the rational data into a multiplex network through simple steps. For any relational data, we want the system could convert it to a multiplex network. On one hand, the system displays the distribution of all attributes in the network, allowing users to understand the data characteristics, and on the other hand provides a rich filtering function, users can filter from any attributes of the data to explore the links between attributes.

This thesis allows us to browse three aspects of the most interesting aspects of the data miming and BigData applied to multimedia archives: The Volume Since our archives are immense and reach orders of magnitude that are usually not practicable for the visualization; Velocity: because of the temporal nature of our data (by definition). The Variety that is a corollary of the richness of multimedia data and of all that one may wish to want to investigate. What we can remember from this thesis is that we met each of these three challenges has been taken in all cases as an answer in the form of a multiplex network analysis. These structures are always at the heart of our work, whether in the criteria for filtering edges using the Simmelian backbone algorithm, or in the superposition of time slices in the complex networks, or that it is much more

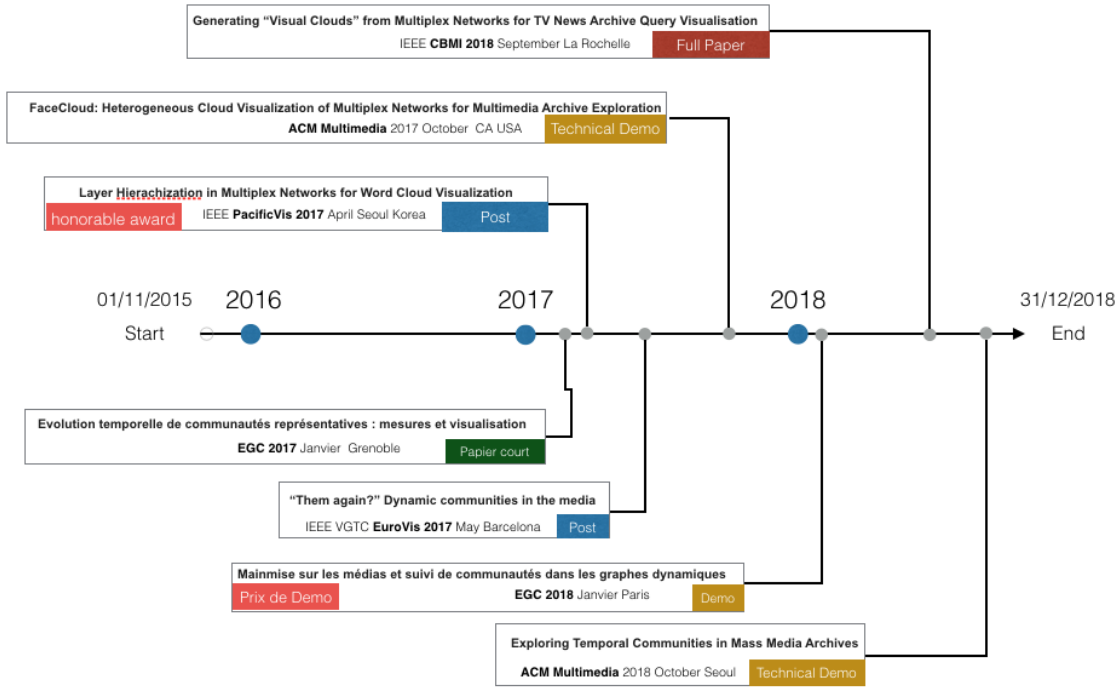
directly in the combinations of visual and textual semantic indices for which we extract hierarchies allowing our visualization.

Publications:

The following list and timeline show the paper we have published during my three years study.

- “Exploring Temporal Communities in Mass Media Archives”, Haolin Ren, Benjamin Renoust, Marie-Luce Viaud, Guy Melançon and Shin’ichi Satoh, ACM MultiMedia (ACMMM), Seoul, South Korea, Oct. 2018
- “Generating Visual Clouds from Multiplex Networks for TV News Archive Query Visualization" Haolin Ren, Benjamin Renoust, Marie-Luce Viaud, Guy Melançon and Shin’ichi Satoh, Content-Based Multimedia Indexing (CBMI), La Rochelle, France, Sep. 2018
- “Mainmise sur les médias et suivi de communautés dans les graphes dynamiques." Haolin Ren, Marie-Luce Viaud, and Guy Melançon, EGC 2018 (pp. 451-454), Paris, France, Jan. 2018
- “FaceCloud: Heterogeneous Cloud Visualization of Multiplex Networks for Multimedia Archive Exploration" Benjamin Renoust, Haolin Ren, Guy Melançon, Marie-Luce Viaud, Shin’ichi Satoh.ACM MultiMedia 2017 Mountain View (CA), USA, Oct 2017
- “Layer Hierarchization in Multiplex Networks for Word Cloud Visualization", Benjamin Renoust, Haolin Ren, Guy Melançon, and Marie-Luce Viaud, IEEE PacificVis 2017, Seoul, South Korea, Apr. 2017
- “‘Them again?’ Dynamic Communities in the Mass Media, Haolin Ren, Marie-Luce Viaud and Guy Melançon, EuroVis 2017, Barcelona, Spain, Jun. 2017
- “Evolution temporelle de communautés représentatives: mesures et visualisation", Haolin Ren, Marie-Luce Viaud and Guy Melançon, EGC 2017 (pp. 417-422), Grenoble, France, Jan 2017

Fig. 6.1 Academic events timeline during my PhD studying shows the paper we have published.



References

[1] Ahn, J. w., Plaisant, C., and Shneiderman, B. (2014). A task taxonomy for network evolution analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):365–376.

[2] Amar, R. and Stasko, J. (2004). A knowledge task-based framework for design and evaluation of information visualizations. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 143–150. IEEE.

[3] Amos, B., Ludwiczuk, B., and Satyanarayanan, M. (2016). Openface: A general-purpose face recognition library with mobile applications. Technical report, Technical report, CMU-CS-16-118, CMU.

[4] Archambault, D., Purchase, H., and Pinaud, B. (2010). Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 17(4):539–552.

[5] Aynaud, T., Fleury, E., Guillaume, J.-L., and Wang, Q. (2013). *Communities in Evolving Networks: Definitions, Detection, and Analysis Techniques*, pages 159–200. Springer New York.

[6] Aynaud, T. and Guillaume, J.-L. (2010). Static community detection algorithms for evolving networks. In *Modeling and optimization in mobile, ad hoc and wireless networks (WiOpt), 2010 proceedings of the 8th international symposium on*, pages 513–519. IEEE.

[7] Bach, B., Henry-Riche, N., Dwyer, T., Madhyastha, T., Fekete, J.-D., and Grabowski, T. (2015). Small multipiles: Piling time to explore temporal patterns in dynamic networks. *Computer Graphics Forum*, 34(3):31–40.

[8] Baeza-Yates, R. and Davis, E. (2004). Web page ranking using link attributes. In *Proc. ACM WWW*, pages 328–329. ACM.

[9] Barry, A. M. (1997). *Visual intelligence: Perception, image, and manipulation in visual communication*. SUNY Press.

[10] Barth, L., Kobourov, S. G., and Pupyrev, S. (2013). An experimental study of algorithms for semantics-preserving word cloud layout. *University of Arizona Report*.

[11] Beck, F., Burch, M., Diehl, S., and Weiskopf, D. (2014). The state of the art in visualizing dynamic graphs. *EuroVis STAR*, 2.

[12] Bekos, M. A., Van Dijk, T. C., Fink, M., Kindermann, P., Kobourov, S., Pupyrev, S., Spoerhase, J., and Wolff, A. (2014). Improved approximation algorithms for box contact representations. In *European Symposium on Algorithms*, pages 87–99. Springer.

[13] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 10:8.

[14] Brehmer, M. and Munzner, T. (2013a). A multi-level typology of abstract visualization tasks. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2376–2385.

[15] Brehmer, M. and Munzner, T. (2013b). A multi-level typology of abstract visualization tasks. *IEEE TVCG*, 19(12):2376–2385.

[16] Brin, S. and Page, L. (2012). Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, 56(18):3825–3833.

[17] Burt, R. and Scott, T. (1985). Relation content in multiple networks. *Social Science Research*, 14:287–308.

[18] Cazabet, R. and Amblard, F. (2011). Simulate to detect: a multi-agent system for community detection. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 02*, pages 402–408.

[19] Cazabet, R., Amblard, F., and Hanachi, C. (2010). Detection of overlapping communities in dynamical social networks. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 309–314. IEEE.

[20] Chakrabarti, D., Kumar, R., and Tomkins, A. (2006). Evolutionary clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 554–560. ACM.

[21] Chen, H., Houston, A. L., Sewell, R. R., and Schatz, B. R. (1998). Internet browsing and searching: User evaluations of category map and concept space techniques. *Journal of the American society for information science*, 49(7):582–603.

[22] Cui, W., Wu, Y., Liu, S., Wei, F., Zhou, M. X., and Qu, H. (2010). Context preserving dynamic word cloud visualization. In *Visualization Symposium (PacificVis), 2010 IEEE Pacific*, pages 121–128. IEEE.

[23] D3js (1999). D3.js.

[24] Erten, C., Harding, P. J., Kobourov, S. G., Wampler, K., and Yee, G. (2003). Graphael: Graph animations with evolving layouts. In *International Symposium on Graph Drawing*, pages 98–110. Springer.

[25] Farrugia, M., Hurley, N., and Quigley, A. (2011). Exploring temporal ego networks using small multiples and tree-ring layouts. *Proc. ACHI*, 2011:23–28.

[26] Fujimura, K., Toda, H., Inoue, T., Hiroshima, N., Kataoka, R., and Sugizaki, M. (2006). Blogranger—a multi-faceted blog search engine. In *Proc. WWW, Weblogging Ecosystem*.

[27] Ghoniem, M., Fekete, J.-D., and Castagliola, P. (2005). On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization (Palgrave)*, 4(2):114–135.

[28] Gilbert, F., Simonetto, P., Zaidi, F., Jourdan, F., and Bourqui, R. (2010). Communities and hierarchical structures in dynamic social networks: analysis and visualization. *Social Network Analysis and Mining*, pages 1–13.

[29] Guimerà, R., Mossa, S., Turtschi, A., and Amaral, L. A. N. (2005). The worldwide air transportation network: anomalous centrality, community structure, and cities' global roles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(22):7794–7799.

[30] Hachul, S. and Jünger, M. (2004). Drawing large graphs with a potential-field-based multilevel algorithm. In *International Symposium on Graph Drawing*, pages 285–295. Springer.

[31] Hearst, M. (2009). *Search user interfaces*. Cambridge University Press.

[32] Hervé, N., Viaud, M.-L., Thièvre, J., Saulnier, A., Champ, J., Letessier, P., Buisson, O., and Joly, A. (2013). Otmedia: the french transmedia news observatory. In *Proc. ACM Multimedia*, pages 441–442. ACM.

[33] Ide *et al.*, I. (2004). Topic threading for structuring a large-scale news video archive. *Image and Video Retrieval*, 1(1):123–131.

[34] Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., and Porter, M. A. (2014). Multilayer networks. *Journal of complex networks*, 2(3):203–271.

[35] Kleiboemer, A. J., Lazear, M. B., and Pedersen, J. O. (1996). Tailoring a retrieval system for naive users. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval (SDAIR), Las Vegas, NV*.

[36] Krackhardt, D. (1999). The ties that torture: Simmelian tie analysis in organizations. *Research in the Sociology of Organizations*, 16(1):183–210.

[37] Krovetz, R. and Croft, B. (1992). Lexical ambiguity and information retrieval. *ACM Trans on Information Systems (TOIS)*, 10(2):115–141.

[38] Kurzhals, K., John, M., Heimerl, F., Kuznecov, P., and Weiskopf, D. (2016). Visual movie analytics. *IEEE TMM*, 18(11):2149–2160.

[39] Lancichinetti, A., Fortunato, S., and Kertész, J. (2009). Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015.

[40] Le, D. D. and Satoh, S. (2011). Indexing faces in broadcast news video archives. In *2011 IEEE 11th ICDM Workshops*, pages 519–526.

[41] LESK, M. A. (1997). *Practical digital libraries: Books, bytes, and bucks*. Morgan Kaufmann.

[42] Marchionini, G. (2006). Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46.

[43] McCormack, G. (2008). Japan and north korea: The long and twisted path toward normalcy. Technical report, Working Paper. US-Korea Inst. at SAIS.

[44] Mitra, B., Tabourier, L., and Roth, C. (2012). Intrinsically dynamic network communities. *Computer Networks*, 56(3):1041–1053.

[45] Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):10.

[46] Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM Review*, 45:167–256.

[47] Ngo *et al.*, T. D. (2013). Face retrieval in large-scale news video datasets. *IEICE Trans. on Information and Systems*, 96(8):1811–1825.

[48] Nick, B., Lee, C., Cunningham, P., and Brandes, U. (2013). Simmelian backbones: Amplifying hidden homophily in facebook networks. In *Advances in Social Network Analysis and Mining (ASONAM)*, pages 525–532.

[49] Noack, A. (2009). Modularity clustering is force-directed layout. *Physical Review E*, 79(2):026102.

[50] Pirolli, P., Schank, P., Hearst, M., and Diehl, C. (1996). Scatter/gather browsing communicates the topic structure of a very large text collection. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 213–220. ACM.

[51] Reiss, A. J. (1959). The sociological study of communities. *Rural Sociology*, 24(2):118.

[52] Renoust, B., Kobayashi, T., Ngo, T. D., Le, D.-D., and Satoh, S. (2016). When face-tracking meets social networks: a story of politics in news videos. *Applied Network Science*, 1(1):4.

[53] Renoust, B., Melançon, G., and Munzner, T. (2015). Detangler: Visual analytics for multiplex networks. *Computer Graphics Forum*, 34(3):321–330.

[54] Renoust, B., Melançon, G., and Viaud, M.-L. (2014). Entanglement in multiplex networks: understanding group cohesion in homophily networks. In *Social Network Analysis*, pages 89–117. Springer.

[55] Riehmann, P., Hanfler, M., and Froehlich, B. (2005). Interactive sankey diagrams. In *IEEE Symposium on Information Visualization*, pages 233–240. IEEE Computer Society.

[56] Robertson, G., Fernandez, R., Fisher, D., Lee, B., and Stasko, J. (2008). Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6).

[57] Rossetti, G. and Cazabet, R. (2018). Community discovery in dynamic networks: a survey. *ACM Computing Surveys (CSUR)*, 51(2):35.

[58] Rosvall, M. and Bergstrom, C. T. (2010). Mapping change in large networks. *PloS one*, 5(1):e8694.

[59] Saket, B., Simonetto, P., and Kobourov, S. (2014). Group-Level Graph Visualization Taxonomy. In Elmqvist, N., Hlawitschka, M., and Kennedy, J., editors, *EuroVis - Short Papers*. The Eurographics Association.

[60] Scaiella, U., Ferragina, P., Marino, A., and Ciaramita, M. (2012). Topical clustering of search results. In *Proc. WSDM*, pages 223–232. ACM.

[61] SemanticUI (2015). Semanticui.

[62] Shavit, A. (2005). The notion of 'group' and tests of group selection. *Philosophy of Science*, 72(5):1052–1063.

[63] Shi, J. and Tomasi, C. (1994). Good features to track. In *Proceedings CVPR'94.*, pages 593–600. IEEE.

[64] Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages*, pages 336–343. IEEE.

[65] Simon, H. A. and Larkin, J. H. (1987). Why a diagram is (sometimes) worth 10,000 words. *Models of Thought*, 2.

[66] Sinclair, J. and Cardew-Hall, M. (2008). The folksonomy tag cloud: when is it useful? *Journal of Information Science*, 34(1):15–29.

[67] Spiliopoulou, M., Ntoutsi, I., Theodoridis, Y., and Schult, R. (2006). Monic: modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 706–711. ACM.

[68] Stacey, R. (2005). Social selves and the notion of the a "group-as-a-whole". *Group*, 29(1):187–209.

[69] Stieglitz, S. and Dang-Xuan, L. (2013). Social media and political communication: a social media analytics framework. *Social Network Analysis and Mining*, 3(4):1277–1291.

[70] Vehlow, C., Beck, F., Auwärter, P., and Weiskopf, D. (2015a). Visualizing the evolution of communities in dynamic graphs. *Computer Graphics Forum*, 34(1):277–288.

[71] Vehlow, C., Beck, F., and Weiskopf, D. (2015b). The state of the art in visualizing group structures in graphs. In *Eurographics Conference on Visualization (EuroVis)-STARs*, volume 2. The Eurographics Association.

[72] Viegas, F. B., Wattenberg, M., and Feinberg, J. (2009). Participatory visualization with wordle. *IEEE TVCG*, 15(6):1137–1144.

[73] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, 57(2):137–154.

[74] Wang, W., Wang, H., Dai, G., and Wang, H. (2006). Visualization of large hierarchical data by circle packing. In *Proc. SIGCHI*, CHI '06, pages 517–520, New York, NY, USA. ACM.

[75] Ware, C. (2000). *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, Orlando, FL.

[76] Ware, C. (2005). Visual queries: The foundation of visual thinking. In Tergan, S.-O. and Keller, T., editors, *Knowledge and Information Visualization*, volume 3426 of *Lecture Notes in Computer Science*, pages 27–35.

[77] Wellman, B. and Leighton, B. (1979). Networks, neighborhoods, and communities: Approaches to the study of the community question. *Urban affairs quarterly*, 14(3):363–390.

[78] Wilson, M. L., Kules, B., Shneiderman, B., et al. (2010). From keyword search to exploration: Designing future search interfaces for the web. *Foundations and Trends® in Web Science*, 2(1):1–97.

[79] Zaidi, F., Muelder, C., and Sallaberry, A. (2014). Analysis and visualization of dynamic networks. *Encyclopedia of Social Network Analysis and Mining*, pages 37–48.

[80] Zhao, Y. and Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. In *Proc. CIKM*, CIKM '02, pages 515–524, New York, NY, USA. ACM.