Journal of Visual Languages and Computing 25 (2014) 433–451

Contents lists available at ScienceDirect

Journal of Visual Languages and Computing

journal homepage: www.elsevier.com/locate/jvlc

# Network visualization for financial crime detection ☆, ☆☆

Walter Didimo, Giuseppe Liotta, Fabrizio Montecchiani *

*Dipartimento di Ingegneria, Università degli Studi di Perugia, Italy*

ARTICLE INFO

*Article history:*
Received 23 July 2013
Received in revised form
18 October 2013
Accepted 10 January 2014
Available online 29 January 2014

*Keywords:*
Financial crime detection
Graph visualization
Social network analysis
Information visualization

ABSTRACT

*Objective:* We present a new software system, VisFAN, for the visual analysis of financial activity networks.
*Methods:* We combine enhanced graph drawing techniques to devise novel algorithms and interaction functionalities for the visual exploration of networked data sets, together with tools for SNA and for the automatic generation of reports.
*Results:* An application example constructed on real data is presented. We also report the results of a study aimed at qualitatively understanding the satisfaction level of the analysts when using VisFAN.
*Conclusion:* VisFAN makes a strong use of visual interactive tools, combined with ad-hoc clustering techniques and customizable layout constraints management.
*Implications:* As this system confirms, information visualization can play a crucial role to face the discovery of financial crimes.



## 1. Introduction

Financial crimes represent a major problem of many governments and are often related to organized crimes like terrorism and narcotics trafficking. Money laundering and frauds are among the most common types of financial crimes. They are based on relevant volumes of financial transactions to conceal the identity, the source, or the destination of illegally gained money. The estimated amount of money laundered globally in 1 year is 2 trillion in current US dollars.[1] To face this problem, most governments have created special investigation agencies, called *financial intelligence units* (FIUs). Financial institutions, like banks, money service businesses, insurance agencies, casinos, must store all transactions executed by their customers in some kind of electronic archive and must report to FIUs about suspicious transactions. The main goal of FIUs is to collect and analyze suspicious transaction data to discover illegal activities, so to defend the integrity of worldwide financial markets and to prevent from organized crimes that can mine the homeland security.

The major challenge of financial subjects and of FIUs is to deal with the volume and the complexity of the collected data, which can be modeled as financial activity networks (FANs) whose nodes represent persons, companies, bank accounts, or other types of entities, and whose edges represent their connections according to a set of possible criteria. It is widely accepted that the exploration of such networks in order to discover criminal patterns strongly benefits from a strict integration of social network analysis (SNA) and visualization tools [17,40,42,43]. Due to the heterogeneity of the data and to the different methodologies followed by financial investigation analysts, any system for the visual analysis of FANs should offer a variety of flexible tools that guarantee strong interaction with the user. As it will be discussed in Section 5, there is a general

☆ This paper has been recommended for acceptance by S.-K. Chang.
☆☆ Research supported in part by the MIUR project AMANDA prot. 2012C4E3KT 001. Extended abstracts on the research of this paper have been presented at the 4th IEEE Pacific Visualization Symposium 2011 and at the IVAPP 2012: International Conference on Information Visualization Theory and Applications.
* Corresponding author.
*E-mail addresses:* walter.didimo@unipg.it (W. Didimo), giuseppe.liotta@unipg.it (G. Liotta), fabrizio.montecchiani@unipg.it (F. Montecchiani).
[1] http://www.unodc.org/unodc/en/money-laundering/globalization.html.

1045-926X/$ - see front matter © 2014 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jvlc.2014.01.002

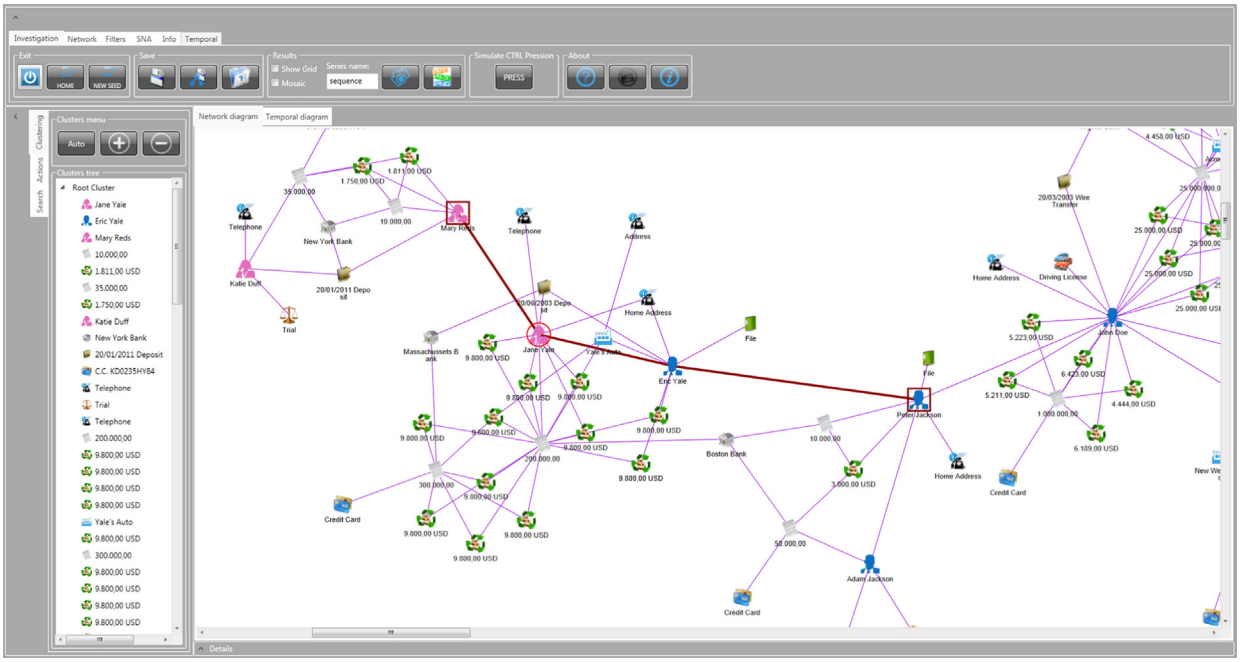


**Fig. 1.** Snapshot of the interface of VisFAN.

lack of these tools in the existing systems for the analysis of criminal networks.

In this paper we present a new system, called VisFAN. A snapshot of the system interface is shown in Fig. 1. A demonstrative video is also available.[2] If compared with previous software and methodologies for criminal network analysis and discovery, VisFAN presents the following main ingredients and novelties:

*Multiple application scenarios*: It allows either the analysis of data within the same financial institution (bank, money service businesses, insurance agency, etc.) or the analysis of suspicious transaction data collected by FIUs from different financial subjects. The system adopts different models to deal with the different types and sizes of the networked data sets managed in these two scenarios. On the other hand, unified algorithms and interaction paradigms are provided for fundamental analysis functionalities, such as clustering and automatic graph drawing.

*Interaction paradigms*: From the data exploration point of view, VisFAN fully combines *bottom-up* and *top-down* paradigms to visually navigate complex networks. Using the bottom-up interaction paradigm the analyst can start from a seminal node and iteratively enhance the network with new elements by adding neighbors of the displayed nodes. With the top-down interaction paradigm the analyst can use hierarchical clustering to recursively aggregate elements on the network currently displayed. Each cluster can be collapsed or expanded independently at any time, so that the analyst can simplify the network at her convenience. These two kinds of interaction paradigms can be arbitrarily alternated while maintaining consistency. A collapsed cluster is represented by a small blue square, whose label can indicate different types of related information.

*Network clustering*: The system makes it possible to mix automatic and manual clustering. It automatically computes a hierarchical clustering by exploiting *k-cores*, which have been proven to be effective for discovering relevant groups in social networks (see, e.g., [2,6,7,22,31,37]). In particular, we propose a general clustering approach and then specialize it to the different kinds of financial activity networks. The analyst can manually edit the current clustering structure at any time by creating new clusters or by modifying the existing ones.

*Geometric constraints*: VisFAN allows the analyst to modify the drawings in several ways. She can easily customize the dimensions of each cluster region; the readability of the drawing inside a cluster region and the number of attributes shown for this drawing are proportional to the dimensions of the region. This kind of interaction can be regarded as a new focus+context

[2] http://youtu.be/OApQMIIryeA.

technique with multiple foci. We remark that experiments providing evidence of the usefulness of focus+context techniques to navigate clustered graphs have been conducted by Schaffer et al. [36]. The analyst can also impose different constraints on the node positions to preserve her mental map. For example, she can force some nodes to stay close together or she can fix the position of a subset of nodes. The drawing algorithm attempts to compute a readable layout while taking into account all the constraints defined by the analyst. To this aim, VisFAN implements a specialized version of the general graph drawing force-directed approach, which is tailored to our needs.

In addition to the ingredients mentioned above, VisFAN is equipped with tools for social network analysis (other than clustering), such as several indexes to measure the centrality of each actor in the network, tools to automatically generate reports, and an advanced mechanism for saving and loading investigations. The VisFAN interface also supports touch functions, so that the user can fully interact with the system and the layouts just using her fingers.

The reminder of this paper is structured as follows. Section 2 describes the characteristics of the financial activity networks that can be analyzed with our system in the different application scenarios. Section 3 presents the functionalities and the algorithmic aspects of VisFAN. The evaluation of the system is presented in Section 4. Section 5 surveys the literature related to our research and discusses the main differences with our contributions. Conclusions and future research directions are discussed in Section 6.

## 2. Financial activity networks

The investigations performed by FIUs are driven by data contained in different types of reports, which are filed by financial institutions like banks, money service businesses, insurance agencies, casinos, etc. There are three main kinds of such reports (see the book by Westphal for further details [42]):

- *Currency transaction report* (*CTR*): It is a report that any financial institution of the United States must file and send to a FIU when one of its customers executes a financial transaction that exceeds 10 000 USD.
- *Suspicious activity report* (*SAR*): It is a report that a financial institution of the United States files and submits to a FIU when it feels that one or more financial transactions executed by some of its customers may be illegal. If the SAR refers to a group of transactions, it reports aggregated data, and many details on each single transaction are lost.
- *Suspicious transaction report* (*STR*): It is a report adopted by many countries in the world. An STR is similar to an SAR but it involves only one transaction and therefore contains more details. If an institution feels that some related transactions are potentially illegal, it has to file and submit to a FIU a different STR for each single transaction.

To complement the generation, collection, and analysis of transaction reports, most governments force every financial institution to record all transactions executed by their customers in some kind of electronic archive; the analysis of these archives should help the institutions to detect suspicious transactions and to report to FIUs about them. Furthermore, a FIU can ask a financial institution to access its electronic archive in order to collect additional data that can be useful for an investigative analysis started by some suspicious transaction reports.

Fig. 2 summarizes the whole process that allows FIUs to collect suspicious transaction data. In the following, we simply refer to STR to denote any kind of suspicious transaction report. In this paper, a *financial activity network* (*FAN*) is either a network that describes relational data coming from a collection of STRs or a network that describes transactions performed by the customers of a specific bank or financial institution. The first kind of FAN is referred to as an *STR network* and its characteristics are described in Section 2.1; the second kind of FAN is referred to as a *banking activity network*, and its characteristics are described in Section 2.2.

### 2.1. STR networks

STRs contain several data, like persons or companies involved in the transactions, amount of the transactions, bank accounts, addresses, types of the transactions, motivations for the transactions, etc. When available, FIUs can complement these data with other kinds of information like, for instance, possible relationships between actors. All data collected by FIUs give rise to STR networks. Therefore, the nodes of an STR network are entities like banks, companies, persons, bank accounts, transactions, and reports filing these data. Links between entities represent many possible types of semantic connections.

For example, Fig. 3 shows a small portion of an STR network. In this network there are five persons and three companies. Persons Peter Brown and Mike Jones are mentioned in an STR document dated 16/02/2011 (icon with exclamation mark); in this document it is also reported a transaction of 50 000 USD performed by Mike Jones. In the layout it is also shown that there is another STR document (dated 22/06/2011) that complements the previous one. The pencil icon represents an investigation task within the FIU that is analyzing these data; this task is also related to a collaboration with the local police (icon with label 111512, indicating the task id), which indicates a connection between Peter Brown and the remaining subjects showed in the layout.

### 2.2. Banking activity networks

Banking activity networks arise from standardized electronic archives maintained by the banks (e.g., the *Archivio Unico Informatico* applies for Italian banks). A FIU

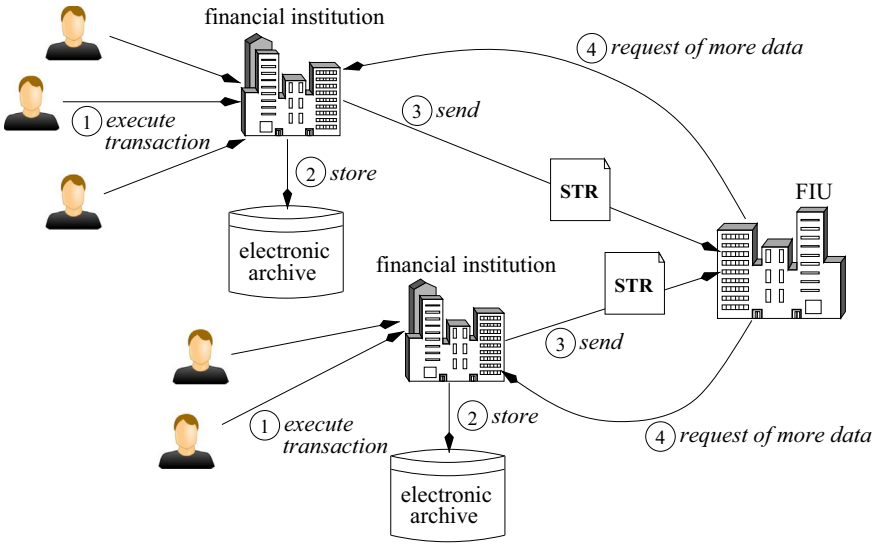


**Fig. 2.** The process by means of which FIUs collect data related to suspicious transaction report.

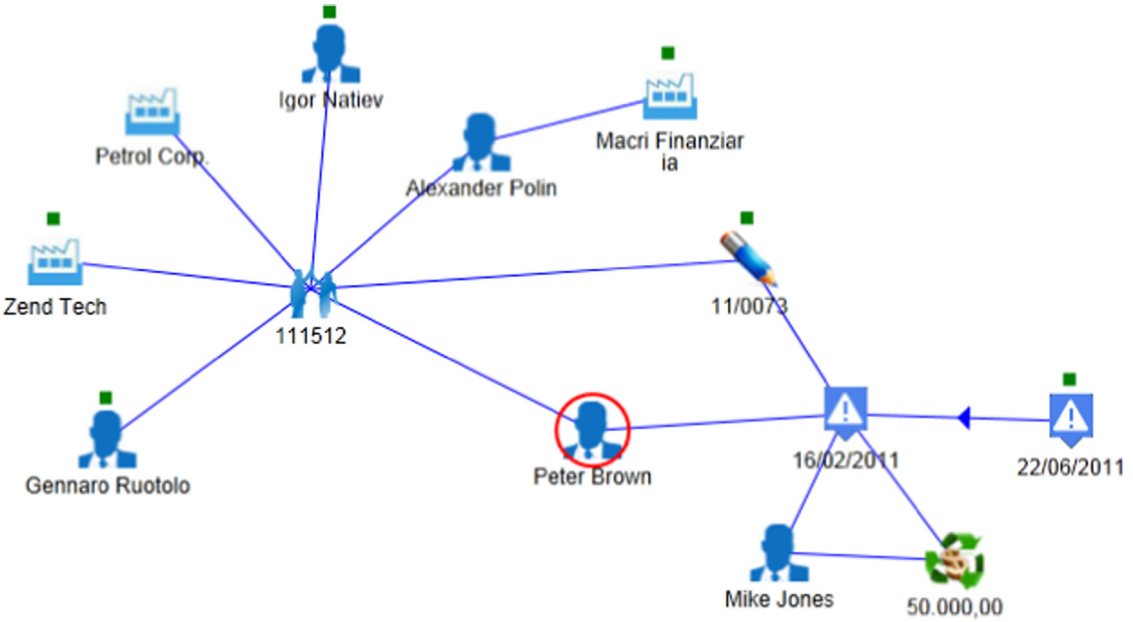


**Fig. 3.** Portion of an STR network.

can access these archives any time during an investigation, in order to acquire more useful information. These archives collect personal data related to the bank's customers, information related to the accounts managed by the bank, as well as every financial transaction involving one of these accounts and exceeding a certain amount of money.

Even in a small time period, a single archive of a medium size bank can record tens of thousands of financial transactions. Thus, from the visualization point of view, it is not helpful to show every single transaction as a single node of the network; instead, a preliminary grouping operation is required. In order to efficiently and effectively handle banking activity networks, we define three types of nodes (see Fig. 4):

- *Subjects*: Physical persons or companies that are customers of the bank.
- *Bank accounts*: A bank account can be linked to several subjects, due to the different roles that a subject can have with respect to a bank account (e.g., owner or delegate).
- *Group of transactions*: Transactions are grouped according to their type (e.g., bank transfer or cash deposit). A node representing a set of transactions has a unique

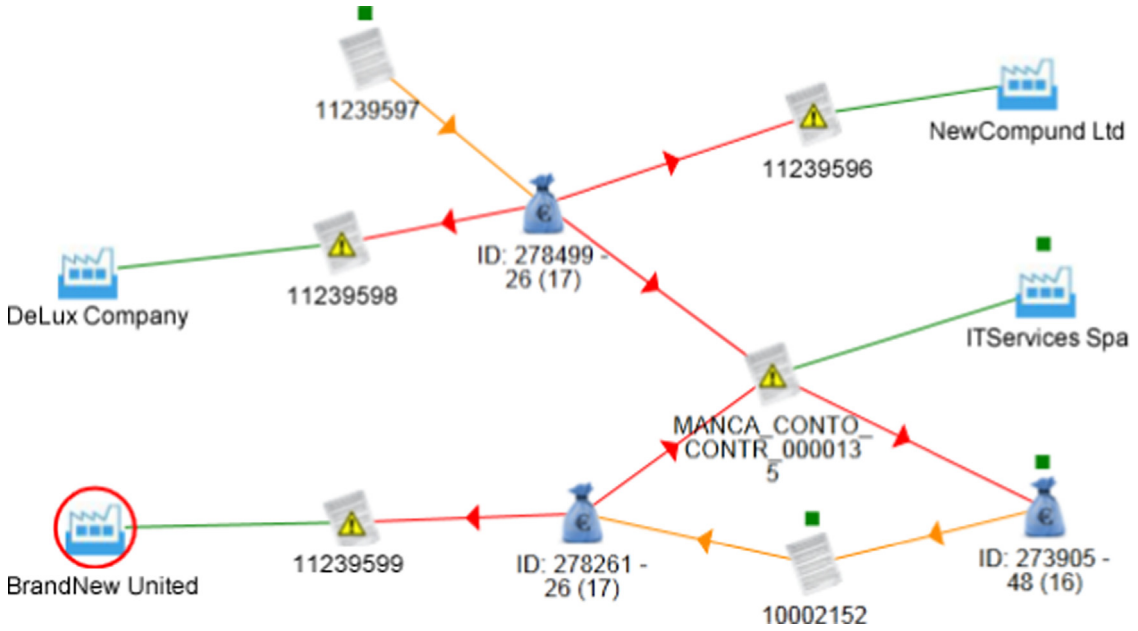


**Fig. 4.** Portion of a banking activity network.

incoming edge from the node representing the source bank account, while it may have several outgoing edges to the nodes representing the recipient bank accounts. The list of transactions grouped in the selected node is showed by the system on user's demand. Together with this list, some related information are displayed, such as statistical data, a line-chart presenting the temporal distribution of the transactions, and a pie-chart displaying their geographical distribution with respect to the bank branches. See also Fig. 5 for an example of this pop-up window.

Fig. 4 shows a subset of bank accounts managed by the same bank; bank accounts are conveyed by the white paper icons (those with an exclamation mark represent accounts for which some data is missing in the electronic log of the bank). In the network are shown four companies operating on some of the accounts (see the undirected edges) and groups of transactions (money bag icons). As mentioned above, transactions in the same group are of the same type, and the corresponding node is labeled with the type's code. The directed edges represent how the money flow among different accounts through groups of transactions.

## 3. The VisFAN system

In this section we describe the system VisFAN. We first discuss the design process of VisFAN in terms of system requirements (Section 3.1); this should help the reader to better understand the motivations behind some of the main functionalities of the system. The architecture and technologies used to design and develop the system are described in Section 3.2. The user interface and the core interaction functionalities of the system are illustrated in Section 3.3. Technical details about the clustering and drawing engines are in Sections 3.4 and 3.5, respectively. Additional functionalities are described in Section 3.6.

### 3.1. System requirements

In order to collect the system requirements we cooperated with analysts of the financial investigation unit of the San Marino Republic, who helped us to understand the typical process followed by FIU analysts. They usually begin a new investigation from a specific STR or from a person involved in an STR, and then explore the network around this seminal node until, based on their experience, they judge the acquired information enough promising to infer potentially criminal patterns, if any. The analyst typically wants to elaborate on the displayed portion of network, and aims to understand how the different actors currently under examination (i.e., those visualized) are related to each other, independently of the other elements of the network. In case the analyst judges the displayed information insufficient, she can continue to expand the network, and both centrality indexes and clusters (if shown) should be updated in the visualization. The system is typically required to visually manage networks that are not too large (up to few hundreds of nodes and edges). Indeed, when the displayed network tends to increase too much, it becomes more and more difficult for the analyst to correctly deal with the visual complexity of the layout; in these cases, the analyst usually decides to save the map and to start from a new investigation seed, possibly comparing different maps later. Hence, the analyst should be allowed to reload at any time all maps generated and saved for the same investigation process, and to use them for comparisons or reporting activities. One important issue is that the system is not required to discover criminal patterns by itself, but it is mainly intended by the analyst as a strong support for

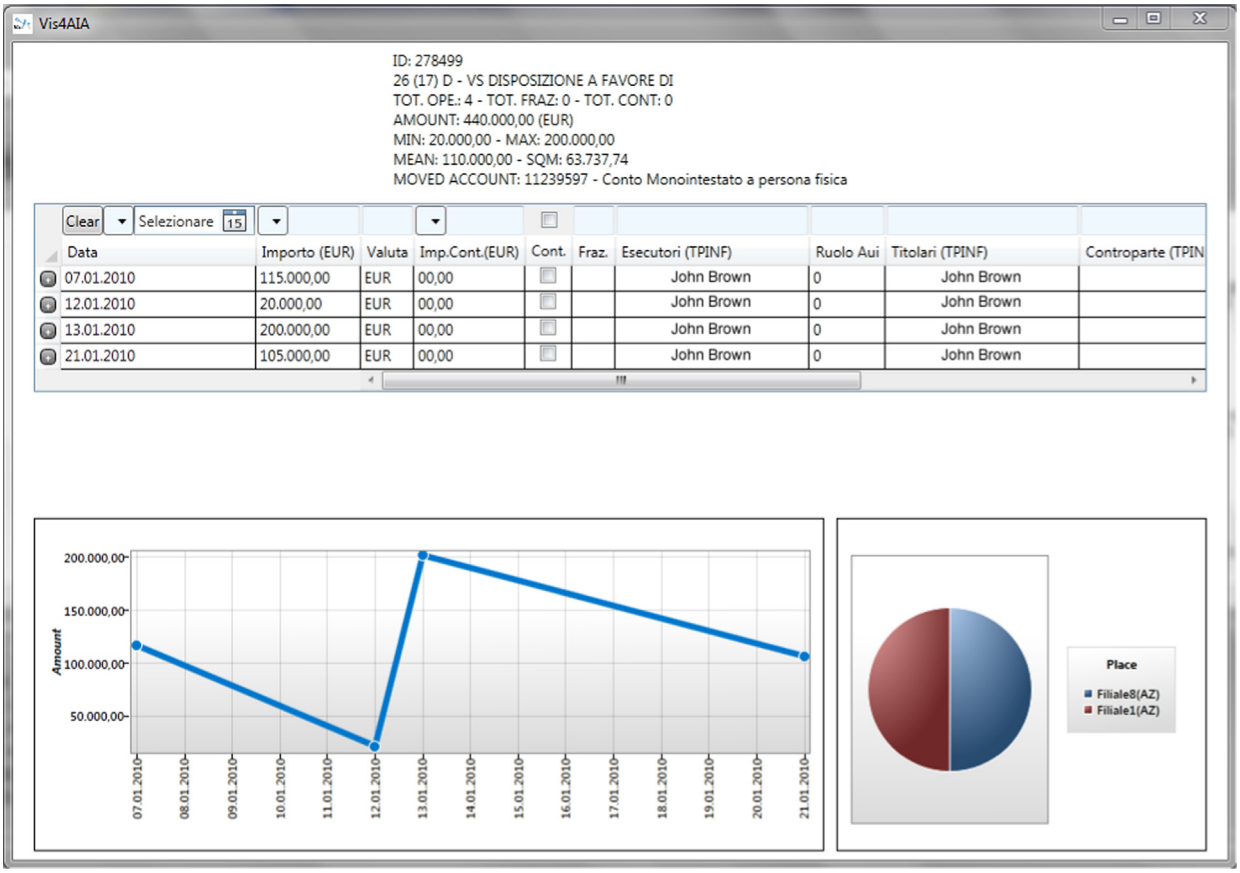


**Fig. 5.** Extra data showed on user's demand for a node of type "group of transactions". The list of transactions along with their temporal and geographical distributions.

the investigation activity. For this reason the system must provide strong interaction, conceived for semi-automatic solutions. For instance, throughout the analysis of the network, the analyst may want to modify previously automatically computed clusters with additional elements or force objects to keep some relative positions in order to preserve her mental map.

### 3.2. Architecture and technologies

The system has been designed to work either in a stand-alone or in a client-server mode. The stand-alone mode is useful in those situations where analysts need to work out of the office, for example in a bank. In such a situation, the analyst may not be able to connect to the Internet. Furthermore, it is often the case that the analyst is not allowed to bring the data out of the bank's premises. In the client-server mode, the data are shared among analysts so they can collaborate and exchange information. The server and the clients communicate through a web service that guarantees the security of the communications via HTTPS. We developed the system exploiting the Microsoft .NET technology and we used C# as programming language. We adopt MySQL as DBMS, both in the stand-alone and in the client-server mode. In terms of software requirements, both the server and the client sides must run under the Microsoft Windows OS (Windows Server 2003 or greater for the server and Windows XP or greater for the client and for the stand-alone mode) and having the Microsoft .NET 4 Full Profile framework installed. In terms of hardware, the Microsoft .NET 4 Full Profile framework requires at least 1 GHz of CPU clock rate and 512 MB of RAM memory. Finally, the client is equipped with a user interface developed exploiting the Microsoft WPF technology and therefore it runs as a desktop application.

### 3.3. Interface and interaction

We designed the visual paradigms and interaction functionalities of VisFAN having in mind the system requirements described above. The analyst can start a new investigation from a desired *seed entity* (e.g., a person, a company, and an STR), searching in the database accessed by VisFAN (see Fig. 6(a)). She can also control the size of the initial network, by specifying the maximum graph theoretic distance from the selected entity (i.e., the maximum number of edges in a shortest path connecting the selected entity to some other entity that will appear in the initial network). With a distance equal to one only the seed entity and its adjacent nodes are displayed. Fig. 6(b) shows an example of an initial network obtained by

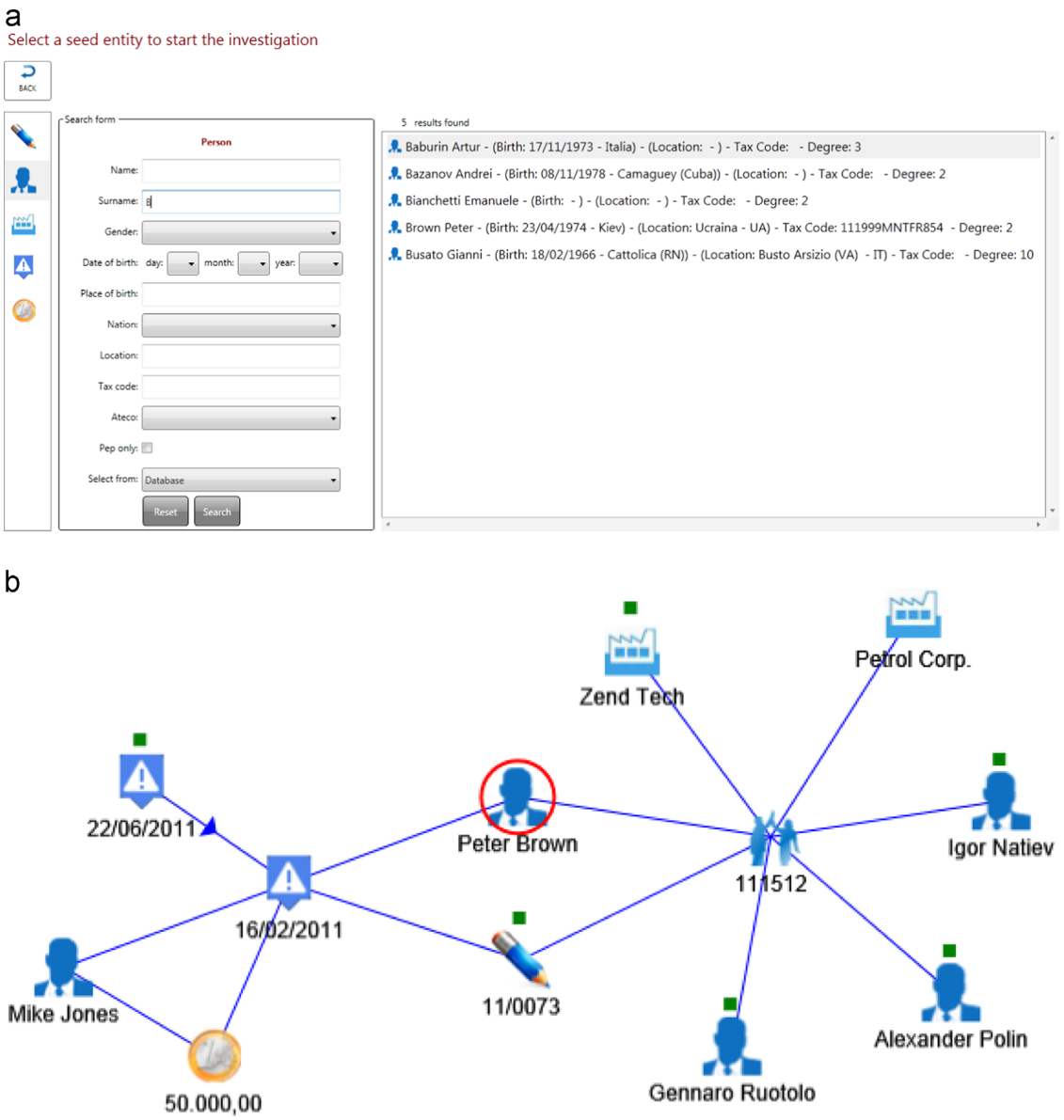


**Fig. 6.** (a) Interface that allows the analyst to begin a new investigation. (b) The initial network obtained from the seed entity *Peter Brown* with maximum graph theoretic distance two. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

considering all entities having graph theoretic distance at most two from the seed entity *Peter Brown* (the selected node in the figure). The analyst can then explore and elaborate on the network with different interactive tools:

*Bottom-up exploration*: The set of nodes and edges can be incrementally enhanced by exploring some of the displayed entities. By double clicking on a node, all of its neighbors and their connections are added to the current network if not already shown. A new layout is automatically computed starting from the status of the previous network, which depends on the presence of cluster regions or other types of geometric constraints. If the previous drawing contained a hierarchy of cluster regions, every node that enters in the new network is automatically assigned to a suitable cluster according to a criterion that will be explained in Section 3.4. A small green box over a node indicates that such a node has not been explored yet; this avoids the analyst to repeat the same exploration action twice. Fig. 7(a) shows the layout of a new network obtained from that of Fig. 6(b) by first exploring entities Alexander Polin and Zend Tech, and then exploring entities Macri Finanziaria and Angelo Macri.

*Top-down exploration*: The analyst can ask the system to automatically compute a cluster hierarchy on the current network. This action will group nodes into clusters and subclusters according to some specific algorithm. Our current clustering algorithm exploits the concept of *k-core*, and it will be explained in details in Section 3.4. Once a cluster hierarchy has been computed, the system decides the initial dimensions of each cluster region based on the number of nodes inside it. In the layout, the boundary of each cluster region is displayed as a rectangle. This seems to be a natural choice since rectangular cluster regions are intuitive for the user, easily support interactive visualizations based on drill-down exploration of the clusters on the computed drawing, and allow for a better control of region area and aspect ratio [18,19].

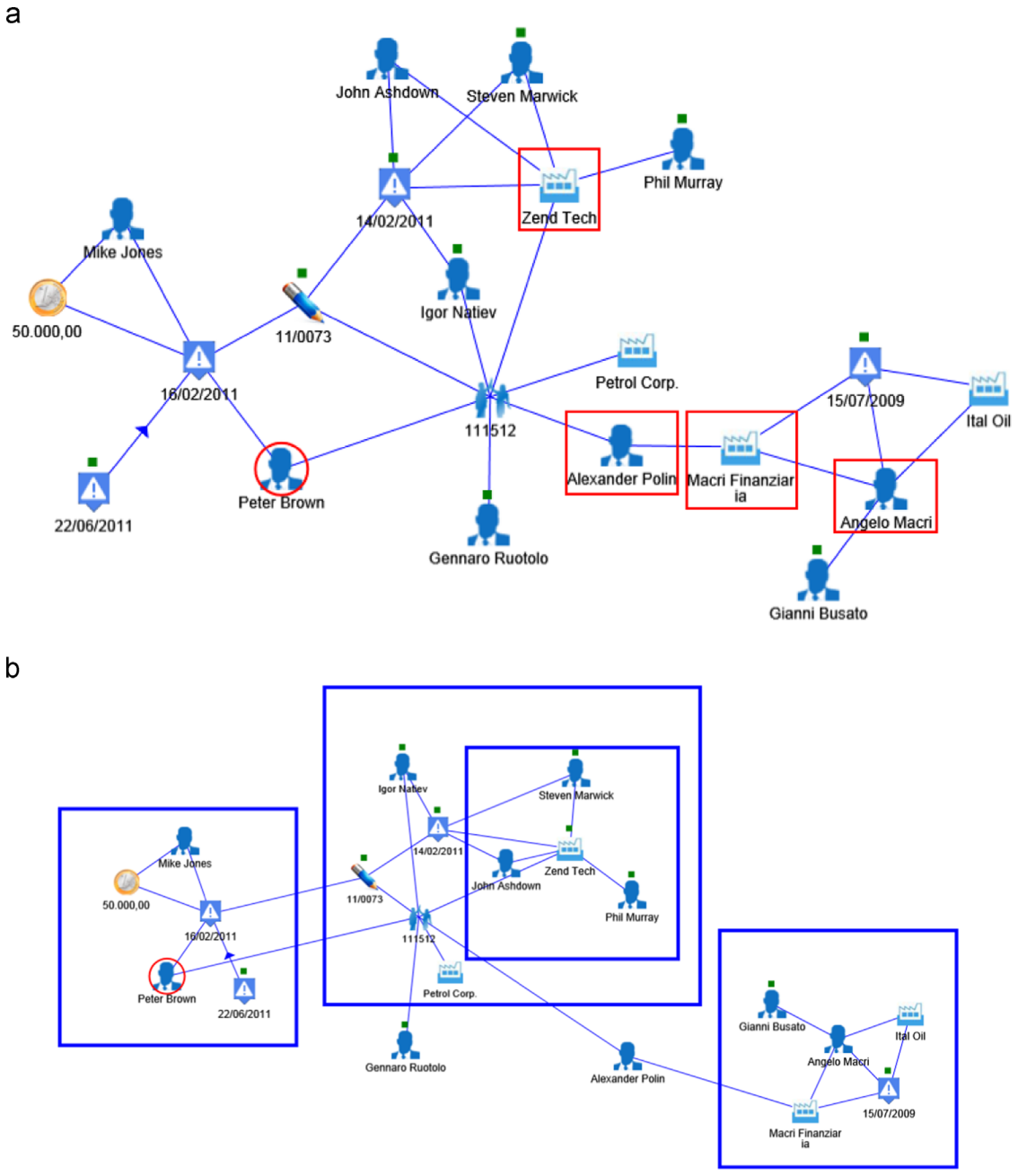


**Fig. 7.** (a) Layout of a new network obtained starting from that of Fig. 6(b) and performing successive bottom-up explorations of some nodes (marked with red boxes). (b) The same network, where some clusters have been defined; each cluster is represented as a rectangular region in the new drawing. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

To help the analyst in capturing the structure of the cluster hierarchy, the corresponding cluster inclusion tree is also displayed on the left-hand side of the interface. VisFAN offers various interaction facilities with the clusters and their regions. The analyst can drag nodes inside or outside a cluster region, so modifying its associated cluster. She can move a cluster inside or outside another cluster, so modifying the cluster hierarchy. She can create new clusters or delete some of the existing ones. She can collapse/expand a cluster region, so to hide/show its interior. The drawing algorithm will react at each user's change in order to rearrange the layout. The analyst can also resize each cluster region at her convenience with the same kind of interaction used to resize a window in a classical operating system graphical user interface, that is, dragging the boundary of the cluster.

Since the drawing algorithm does not allow overlap between clusters that have not inclusion relationships, the readability of the drawing inside a cluster region and the number of attributes shown for this drawing are proportional to the dimensions of the region. Resizing clusters acts as a new focus+context technique with multiple foci. Fig. 7(b) shows a new layout of the network in Fig. 7(a), where some clusters have been defined. In Fig. 8 a cluster region has been collapsed and others have been resized. The label associated with the blue square representing a collapsed cluster can indicate different kinds of information concerning the collapsed cluster. The current version of the system shows the size of the cluster in terms of its number of vertices (see Fig. 8). Alternatively one could label a collapsed cluster with the label of the most central actor in the cluster, according to a desired centrality index. Fig. 9 depicts the same network, after several bottom-up steps and the application of the clustering algorithm.

*Geometric constraints and filtering*: The analyst can apply several constraints on the position of the nodes during the exploration of the network. Using the mouse she can select a group of vertices and ask the system to keep them close or far to each other, or to stay near to a selected point of the canvas; she can also fix the position of a subset of nodes. The drawing algorithm attempts to compute a readable layout while taking into account all the constraints defined by the analyst. The use of geometric constraints can help the analyst to keep her mental map during the browsing of the network and to have a certain control on the relative positions of some nodes based on their typology. Furthermore, the analyst can filter out any desired type of entities and/or connections to reduce the visual complexity of the layout.

*Node* VisFANimplements a wide range of node

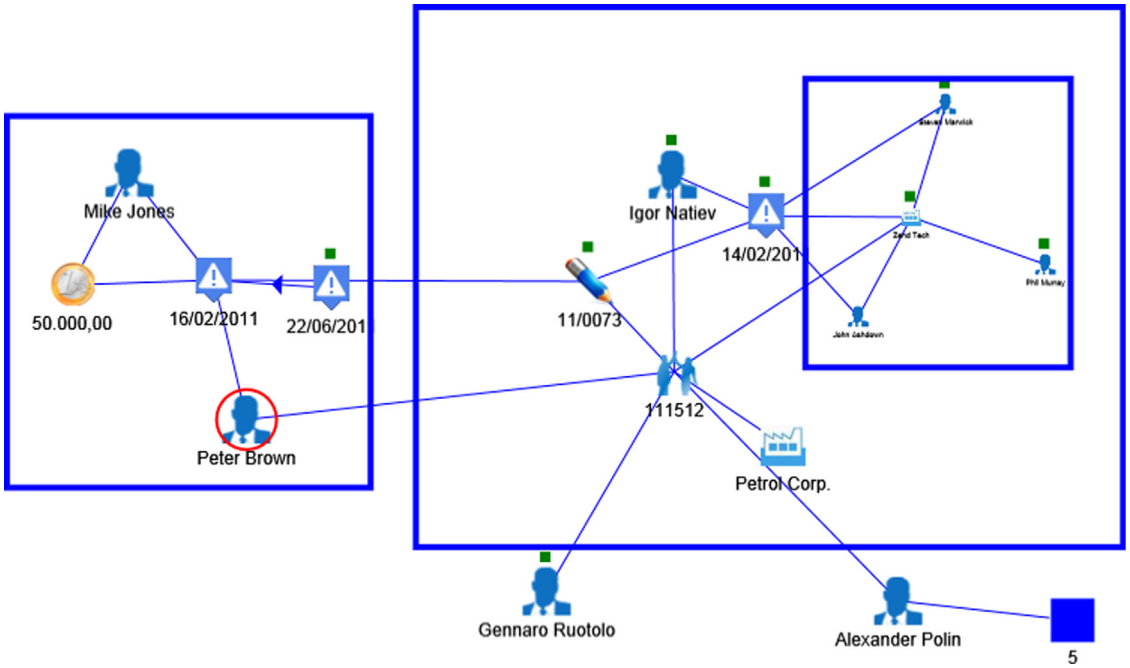


**Fig. 8.** A different layout of the clustered network of Fig. 7(b) where a cluster has been collapsed (the filled node) and another has been resized. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

*centrality*: centrality indexes, like betweenness, closeness, degree, eigenvector centrality [27,35], and the most recent information centrality [33]. The value of a specific type of index is visually conveyed by a small colored circle near to its associated node (different colors are used to denote the range in which the index value falls). The analysts can quickly switch from the visualization of a type of index to another, and all indexes are normalized to be easily compared. The use of geometric constraints can be combined with that of node centrality indexes; for instance, the analyst can decide to fix the position of the most relevant vertex in the center of the canvas. We remark that centrality indexes make sense only for nodes representing the actors of the network, like persons, companies, and bank accounts (see also Section 3.4 for a high-level classification of the different types of nodes in our financial networks). Hence, in order to compute the centrality of these actors, we run the algorithm on a different suitable network consisting of actors only. Namely, we connect two actors if the length of the shortest path between them is at most $d$ (for a preset constant $d$), and then we remove all nodes that are not actors. Fig. 10(a) shows the indexes of degree centrality displayed as node attributes in the drawing. In this case, we fixed $d=2$.

### 3.4. Clustering engine

Our automatic clustering algorithm relies on the concept of *k-core*, which was introduced by Seidman in 1983 [37], and successfully used in social network analysis (see, e.g., [2,6,7,9,22,31]). In the following we recall the definition of $k$-core and describe how the clustering algorithm works.

**Definition 1.** Let $G=(V,E)$ be a (multi)graph and $k$ be an integer number such that $0 \le k < |V|$. Let $W$ be a non-empty subset of $V$. The induced subgraph $G[W]$ of $G$ is a *k-core* (or a *core of order k*) of $G$ if all nodes of $W$ have degree at least $k$ in $G[W]$ and $G[W]$ is a maximal subgraph of $G$ with this property.

Due to the maximality property, if a (multi)graph $G$ contains a $k$-core, such a $k$-core is unique. Note that the $k$-core of $G$ is not necessarily connected.

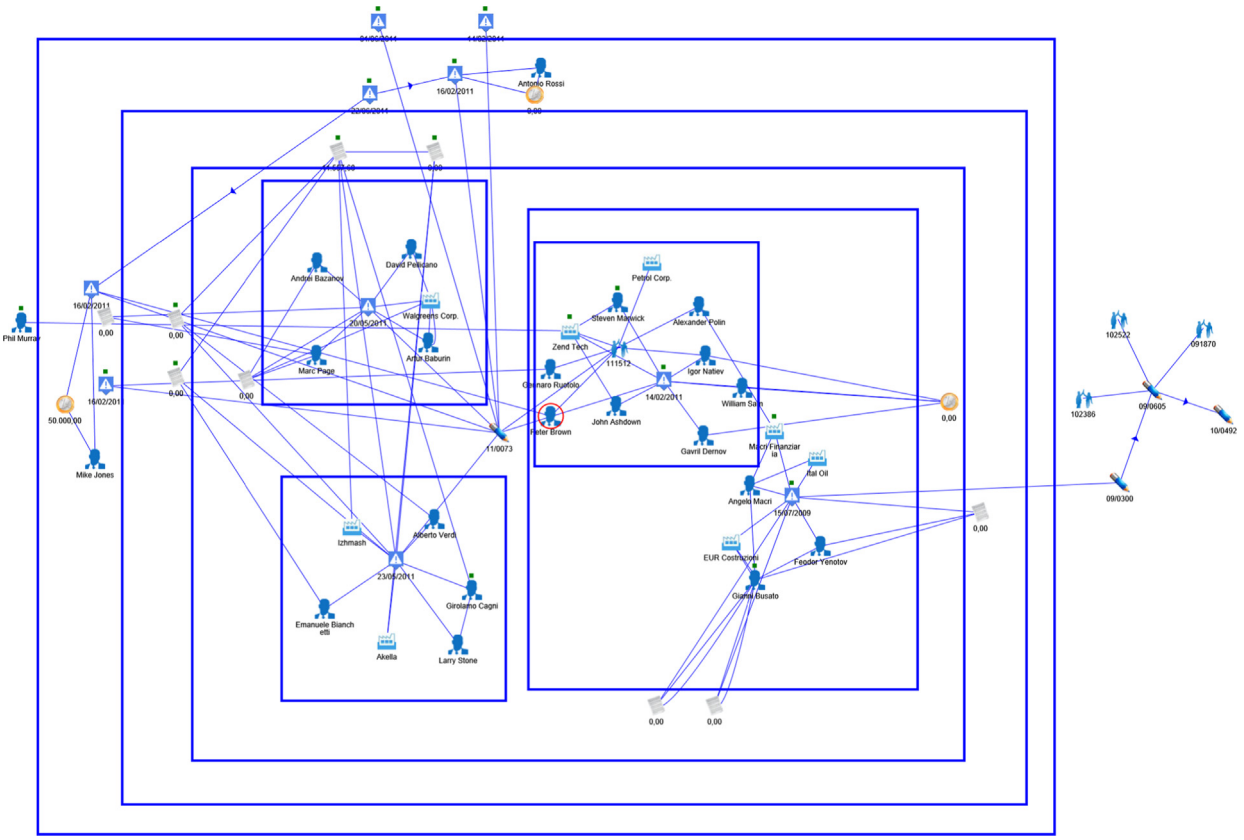


**Fig. 9.** The same network in Fig. 8, after several bottom-up steps and the application of the automatic clustering algorithm. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

**Definition 2.** A connected component of a $k$-core of $G$ is called a *$k$-core component* of $G$.

**Definition 3.** The *core number* of a node of a (multi)graph $G$ is the highest order of a core that contains this node.

For any fixed $k$, denote by $G_k$ the $k$-core of $G$ (if it exists). An immediate consequence of Definition 1 is that if $G$ contains a $k$-core $G_k$, for $k \geq 1$, then $G$ contains a $(k-1)$-core $G_{k-1}$, and $G_k \subseteq G_{k-1}$ ($G_k$ and $G_{k-1}$ might coincide). This because, each node of $G_k$ has degree at least $k$, and therefore it has degree at least $k-1$. This also implies that a $k$-core is the subgraph induced by all nodes with core number at least $k$. For example, Fig. 11(a) shows a graph $G$ where the nodes are labeled with their core numbers. The different $k$-core components of $G$ are represented using closed regions; regions with the same color belong to the same $k$-core and darker colors correspond to cores of higher order. The graph in the figure has four cores: the 1-core (region A), the 2-core (region B), and the 4-core (region E) have one connected component only, while the 3-core has two connected components (regions C and D). Clearly, the 1-core coincides with the whole graph, since the graph does not contain isolated nodes.

For a given graph $G$, our clustering algorithm computes all $k$-cores of $G$, and defines a distinct cluster for each $k$-core component. The inclusion relationships between clusters correspond to the inclusion relationships between the associated $k$-core components. Fig. 11(b) shows the inclusion tree of the cluster hierarchy defined by our algorithm on $G$, based on its $k$-core decomposition. In order to compute all clusters and their inclusion relationships, the clustering algorithm applies the following steps:

*Step 1*: It computes the core number of each node of $G$. This computation can be easily done in $O(n+m)$ time by using, for example, the algorithm in [8], where $n$ and $m$ are the number of nodes and edges of $G$, respectively.

*Step 2*: Denote by $K$ the maximum core number of a node. For each $k = 1, \ldots, K$, the $k$-core components of $G$ are computed. Namely, each $k$-core $G_k$ is the subgraph of $G$ induced by all nodes with core number at least $k$; the $k$-core components of $G_k$ correspond to the different connected components of $G_k$, which can be computed in linear time with a breadth-first or a depth-first search of $G_k$. Denote by $C_k$ a connected component of $G_k$; the nodes of $C_k$ define a new cluster in $G$ and, if $k > 1$, the father node of this cluster in the inclusion tree will be the tree node associated with the $(k-1)$-core component that contains $C_k$ in the graph. Hence, all clusters and their inclusion relationships are computed in linear time by using the node core numbers computed in the previous step, and going recursively in the structure of the different $k$-core components.

From the description of the clustering algorithm, it follows that its time complexity is $O(n+m)$, where $n$ is the number of nodes of $G$ and $m$ is the number of edges of $G$.

We now discuss a further improvement of our clustering approach. Let $G$ be a graph obtained by the analyst after some bottom-up exploration steps. Due to the nature of our financial activity networks, $G$ typically consists of several nodes, some of them being the central actors of the network (persons, companies, or bank accounts), others being their attributes (e.g., addresses), and some others that establish direct or indirect relationships between actors, like nodes representing STR or financial transactions. We refer to the three types of nodes mentioned above as *actor nodes*, *attribute nodes*, and *structure node*,

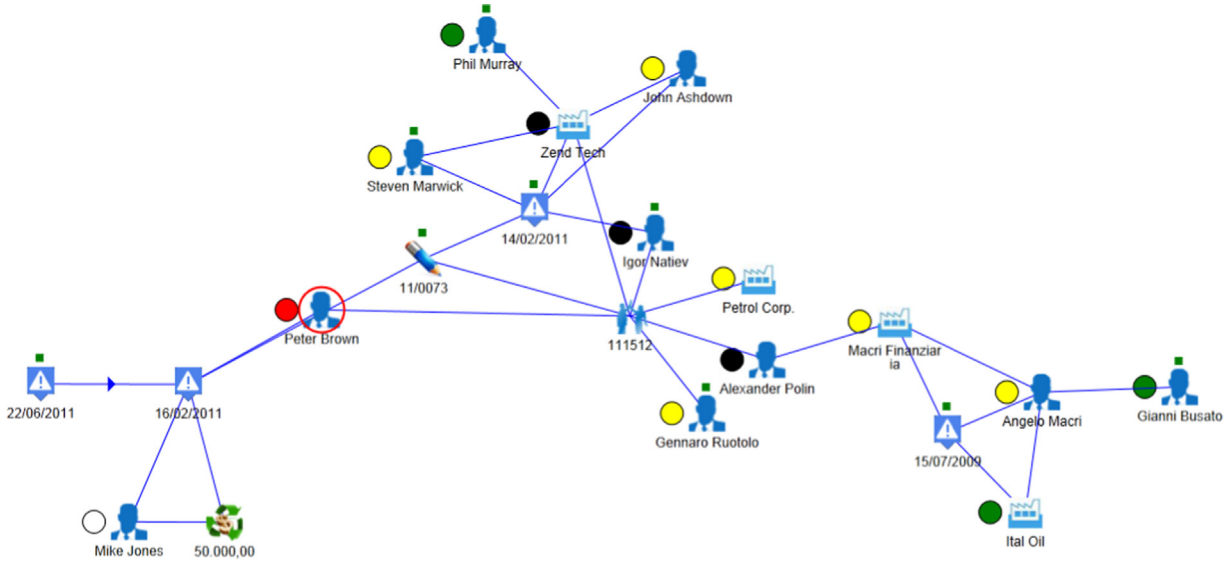


**Fig. 10.** A FAN and the degree centrality levels of its nodes shown as colored disks. White, green, yellow, red, and black disks indicate increasing centrality values. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

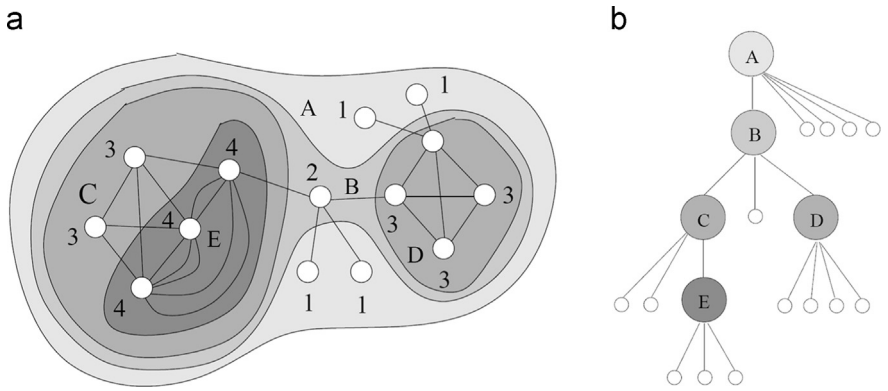


**Fig. 11.** (a) A Graph $G$ where the nodes are labeled with their core numbers. The closed regions colored with different gray scales denote the $k$-core components, and each region is labeled with a different letter. (b) The inclusion tree of the cluster hierarchy defined by our algorithm on graph $G$.

respectively. Most nodes in $G$ have typically very low degree; for example, an attribute typically has degree one and a transaction typically involves a couple of nodes. Hence, applying our clustering algorithm directly on $G$ usually leads to few clusters representing 1-core or 2-core components. Also, attribute nodes of degree one are always assigned to 1-core components, and consequently they could be placed very far from their associated actors in the final drawing. At least for an attribute node it is reasonable that it is assigned to the same cluster of its associated actor, so that they stay close together in the layout. Therefore, in order to get more effective clusters, which are mainly representative of critical patterns, we run our algorithm on a slightly different network $G'$, obtained from $G$ with the following transformation operations:

*Op 1.* Attribute nodes of degree one, if any, are temporarily removed.

*Op 2.* For each structure node $v$, each pair of actor nodes adjacent to $v$ is connected with a new edge. This may give rise to multiple edges.

Once the clustering algorithm is applied to $G'$, we remove the additional edges added with Op 2 and we reinsert attribute nodes of degree one removed with Op 1, by assigning them to the same cluster of its adjacent node. Finally, since structure nodes act as a sort of hubs between their neighbors, we reassign each structure node $v$ to the cluster that is the lowest common ancestor of all neighboring actor nodes of $v$ in the inclusion tree defined by the cluster hierarchy; this seems to be a natural choice to visually convey the role of $v$ with respect to its neighboring actor nodes.

Fig. 12(a) shows an example of an initial network $G$, with one attribute node (the dashed one), several actor nodes (in white), and some structure nodes (in black). Fig. 12(b) shows the network $G'$ obtained with the transformation Op 1 and Op 2; Fig. 12(c) shows the output of the clustering algorithm on $G'$ and the reinsertion of the degree-one attribute nodes; finally, Fig. 12(d) depicts the result once the clusters of the structure nodes have been redefined.

We finally remark that although applying the clustering algorithm on the transformed graph $G'$ (instead that on $G$) leads to clusters that better reflect the hierarchical structure of potential criminal patterns, it takes in general more computational time, since the number of edges of $G'$ is in general higher than that of $G$ of an $O(n)$ factor.

### 3.5. Drawing engine

The drawing algorithm implemented in VisFAN is a force-directed technique whose basic physical model is inspired to that proposed by Fruchterman and Reingold [29]. Similarly to the approach originally proposed by Eades [23], in the model of Fruchterman and Reingold nodes behave as electric particles connected by springs; the electrical charges cause repulsion between nodes, while the springs cause attraction. In the model proposed by Fruchterman and Reingold, given two nodes $u$ and $v$, the attractive force $F_a$ and the repulsive force $F_r$ between $u$ and $v$ are modeled by the following equations (which only approximate a realistic physical model):

$$F_a(u,v) = \frac{d^2(u,v)}{k}, \quad F_r(u,v) = -\frac{k^2}{d(u,v)}$$

where $k$ is a constant that represents the desired distance between any two nodes, and $d(u,v)$ is the real distance between $u$ and $v$. In the implementation of Fruchterman and Reingold repulsive forces act only between adjacent nodes, so to reduce the running time of the algorithm. Like every classical force-directed method, the algorithm starts from an initial position of the nodes and then iteratively moves the nodes according to the forces acting on them; the algorithm stops after a preset number of iterations or when a suitable total energy function of the system becomes sufficiently small.

In order to deal with clusters and other constraints, our physical model is slightly modified and equipped with extra objects (see Sections 3.5.1 and 3.5.2). We use the following more general equations for the forces:

$$F_a(u,v) = K_a(u,v)\frac{d^2(u,v)}{k}, F_r(u,v) = -K_r(u,v)\frac{k^2}{d(u,v)}$$

where $K_a(u,v)$ and $K_r(u,v)$ are real values that depend on the constraints defined on $u$ and $v$, and are used to

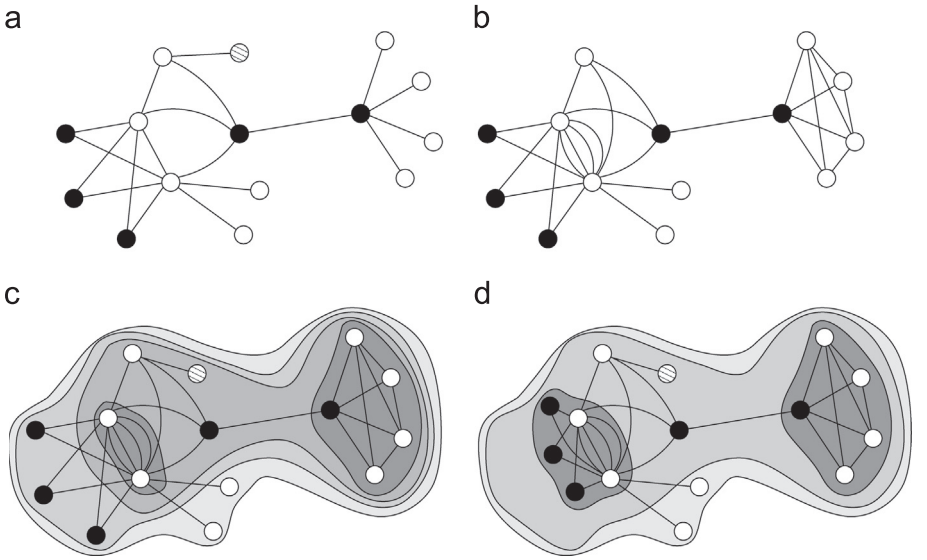


**Fig. 12.** (a) An initial network with actor nodes (white nodes) and structure nodes (black vertices) and one attribute node (the dashed nodes). (b) The new network obtained by creating cliques among all actors adjacent to the same structure node. (c) The $k$-core clustering obtained on the new network (with the subsequent assignment of the attribute node); the edges added artificially to the original network will not appear in the drawing. (d) The final clustering, where the clusters of some structure nodes have been redefined.

increase or to reduce the intensity of the forces, as clarified in the next subsections. If there is no constraint on $u$ and $v$, then $K_a(u,v) = K_r(u,v) = 1$, and we fall in the model of Fruchterman and Reingold.

Since the number of iterations of our force-directed drawing algorithm is bounded above by a constant, which we tuned experimentally, its time complexity is quadratic in the number of vertices of the graph. For example, the drawing algorithm takes in the average 6 s for clustered graphs with 100 vertices and 300 edges on a common laptop with an i5 Intel processor, 2.26 GHz, and 8 GB RAM; it requires about 2 min for graphs that are four times larger, both in terms of nodes and in terms of edges. As it will be discussed in Section 5, the model used by our drawing algorithm is mainly related to that of Eades and Huang [25], for which no running time analysis is provided. Another force-directed algorithm used for drawing clustered graphs is that described by Dogrusoz et al. [21]; although the asymptotic time complexity of that algorithm is the same as our algorithm, an experimental analysis given in [21] reveals that its practical running time is lower than ours (it takes about 2 s for a graph with 100 vertices and 150 edges, on a Pentium IV with 2 GHz and 512 MB RAM). This depends on the higher number of additional constraints (dummy objects and forces) that we add to the physical model for the sake of interaction (see Sections 3.5.1 and 3.5.2); on the contrary, the algorithm in [21] is not conceived for interaction (as discussed in Section 5).

### 3.5.1. Handling clusters

Similar to the idea of Eades and Huang [25], for each cluster $C$ we define a dummy node $v_C$ placed at the center of the rectangular region representing $C$, and connect $v_C$ to all nodes belonging to $C$. Then, we increase the intensity of the attractive force acting between $v_C$ and any other node of $C$ by setting $K_a(v_C,u) > 1$ for each $u$ in $C$. Also, we reduce the intensity of the repulsive force between any two nodes $u$ and $v$ of $C$ by setting $K_r(u,v) < 1$. The value assigned to $K_r(u,v)$ depends on the dimensions of the cluster region. Smaller clusters will determine smaller values of $K_r(u,v)$. This strategy helps to keep the nodes of $C$ relatively close to each other. However, it may still happen that either a node of $C$ leaves the rectangular region representing $C$ or that a node not belonging to $C$ enters in such a region. To avoid these situations, we enhance our physical system by modeling the boundary of each cluster region as a heavy rigid body; if a node $u$ hits such a boundary on a side $s$, we stop the movement of $u$ in the direction orthogonal to $s$. This prevents $u$ from entering or leaving the cluster region. Also, as for nodes, we allow cluster regions to move in the drawing, although they oppose a certain resistance to the forces acting on them. More precisely, we move a cluster region when there are some nodes that hit its boundary (from inside or from outside). We compute the resultant of all forces exerted by these nodes on the boundary and we move the cluster region along the direction of this resultant, after a suitable reduction of its intensity; this reduction is done to simulate the inertia of a heavy rigid body (see Fig. 13). Once the position of a cluster region is updated, a further step is executed to readjust the position of those nodes that might have improperly traversed the boundary of their cluster regions. With a similar strategy we handle the situations when a cluster region hits another cluster region, so avoiding that two cluster regions overlap.

### 3.5.2. Handling relative and absolute node positions

Constraints on the node positions can be handled easier, varying the values of functions $K_a$ and $K_r$. If we impose a constraint that forces a group $U$ of nodes to stay close together in the drawing, the algorithm adds the minimum number of dummy edges required to make $U$ a clique, and then strongly reduces the intensity of the repulsive force acting between every pair of nodes $u$ and $v$ of $U$. This is done by setting

$K_r(u,v) \leq 0.5$. On the contrary, if we set a constraint that forces the vertices of $U$ to stay far from each other, the intensity of the repulsive force acting between every pair of nodes $u$ and $v$ of $U$ is strongly increased, by setting $K_r(u,v) \geq 2$. More suitable values for $K_r(u,v)$ can be tuned experimentally. To fix the position of a node $u$ in the drawing, the algorithm cancels the effect of all forces exerted by every other node on $u$. This is done by setting $K_a(u,v) = K_r(u,v) = 0$, for every vertex $v \neq u$. If we wish that a certain group $U$ of nodes remains relatively close to a certain point $p$ on the canvas, the algorithm creates a dummy node $u$ fixed at location $p$, and connects $u$ to all nodes of $U$ using dummy edges.

### 3.6. Additional functionalities

An investigation can last several weeks or even months; in this period the analyst may want to investigate many companies and persons, eventually deciding if they are involved or not in a financial crime. Thus, the system should provide a saving system able to support the user in creating branches easily, as well as reverting to a previous save. VisFAN provides a tree-based saving system that allows the user to create several branches and to start back from any previous save (see, e.g., Fig. 14).

Once the analyst has found some evidence against a subject, the next step is to present the results of her analysis to a judge. In this situation, providing a detailed documentation of the performed investigation is a key factor. To this aim, VisFAN offers functionalities to generate reports in different ways and formats. Namely

- *Images*: Snapshots of the displayed networks can be saved and classified. In order to facilitate the printing, the system can snap a grid on the canvas and save each single grid cell in a different file. Hence, the user can manually adjust the layout of the network, in order to divide it into a desired number of files. The size of each grid cell can be defined by the user or can automatically computed to fit some standard paper format.
- *Presentations*: The system can automatically generate a presentation of the investigation and its entities, according to some predefined graphical template. The details of every single entity in the network are reported in a different slide of the presentation. Currently, VisFAN supports the Microsoft Power Point format (Office 2010).
- *Spreadsheets*: The system can automatically generate spreadsheets that report details on to the entities of a displayed network. Currently, VisFAN supports the Microsoft Excel format (Office 2010).

It is worth observing that although the Microsoft Office 2010 Suite is quite diffused, some public offices of several countries (like Italy or other European countries) are required to use open source/free software. This issue can be partially overcome by exporting in PDF format, which is a feature

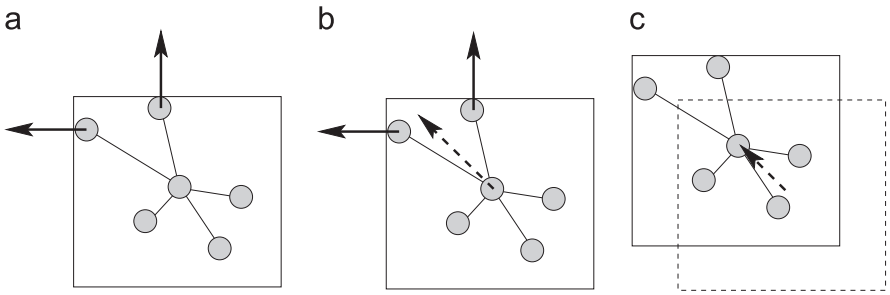


**Fig. 13.** Schematic illustration that shows how a cluster is moved when some nodes hit its boundary. (a) Two nodes that hit the boundary of a cluster from the inside; the arrows represent the corresponding forces exerted by two nodes on the cluster region. (b) The resultant force depicted as a dashed arrow. (c) The movement of the cluster region; the intensity of the resultant is decreased to simulate the inertia of the cluster region.

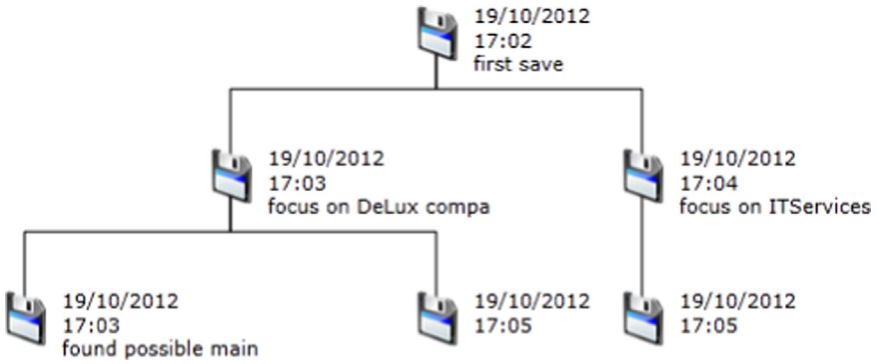


**Fig. 14.** An example of a tree of saves.

available in the Microsoft Office 2010 Suite. In the near future, we plan to support the OpenDocument format (ODF).

We finally remark that the VisFAN interface supports touch functions, so that the user can fully interact with the system and the layouts just using her fingers.

## 4. Case study and user evaluation

AIF has used VisFANfor about 2 years, since its prototype versions. For legal reasons, AIF did not allow us to publish statistics about the usage of the system. For the same reason, we are not allowed to publish real data owned by AIF, even with randomized names and references. However, in Section 4.1 we present an application example constructed on real data described in an official document published by the Financial Crimes Enforcement Network (FinCEN)—United States Department of the Treasury in the 2003 [26]. Furthermore, AIF agreed on answering a questionnaire aimed at qualitatively understanding its satisfaction level with respect to VisFAN; the results of this evaluation are presented and discussed in Section 4.2.

### 4.1. Application example

We present an example of application of our system to a financial activity network that contains two fraudulent patterns. These patterns correspond to real data described in an official document[3] published by the Financial Crimes Enforcement Network (FinCEN)—United States Department of the Treasury in the 2003 [26]. In this document, each real fraudulent pattern is described using similar (dummy) names for the subjects; we partially redefined these names to have more distinguishable entities in the drawing.

Following the narratives contained in that document, we were able to reconstruct the corresponding networks in our system, and we embedded these networks in a connected super network containing extra elements artificially created to add some noise to the fraudulent patterns. As extra elements we imagined new people representing relatives of the real actors, some transactions performed by these persons, and some other attributes like addresses, documents, credit cards, etc. The real data consist of 68 nodes and 112 edges. The artificial data consist of 36 nodes and 51 edges. The two fraudulent patterns refer to two distinct *structuring schemes*, where one or more subjects perform a sequence of small transactions for money laundering (refer to the book of Westphal for more details on this kind of pattern [42]). Fig. 15 shows a picture of the network with both all real and artificial elements. This network was obtained after a few bottom-up exploration steps starting from person *Jane Yale* (the leftmost node of the drawing marked with a red circle), and it contains 104 elements and 163 connections. In the graphical representation of the network, the bank accounts are conveyed by white document icons (labeled with dummy amounts, since this information was not contained in the FinCEN's documentation).

On the displayed network, we applied our clustering algorithm; a drawing of the resulting clustered network is shown in Fig. 16. The computation of the clustering took 0.05 s, while the computation of the drawing took 7 s. Indeed, the most demanding task is the computation of the drawing, whereas clustering does not affect too much the overall time requirement. The cluster hierarchy computed by our algorithm reflects quite carefully the patterns described by FinCEN's document, thus providing some valuable hints to the analyst. More precisely, from the picture one can observe that there exist two disjoint clusters, each containing other subclusters. The disjoint clusters capture the two distinct fraudulent patterns, and the deepest clusters inside each of them include, or they are very close to, the actors (persons and bank accounts) mainly involved in the illegal activity. The deepest cluster on the right-hand side of the layout contains person John Doe (marked with a red circle) and two bank accounts connected to several transactions executed by him; John Doe corresponds to one of the guilty persons in the FinCEN's document. The deepest cluster on the left-hand side of the layout contains two bank accounts, which are connected to each other by several transactions. Also, it is possible to see that one of these two accounts has several connections with a subset of transactions performed by Jane Yale (marked with a red circle), who was also one of the guilty persons in the FinCEN's narrative.

In Fig. 17 the closeness centrality levels of all actors are shown using colored circles. Yellow, red, and black disks represent the highest values of centrality, in this increasing order. From this information, the analyst can get additional hints. For example, the black disk near to John Doe confirms his centrality in the illegal activity. Also, there is a person, Erik Yale, directly connected to Jane Yale whose centrality value is rather high (in the picture this person is marked with a red circle and its level of centrality is indicated by a red disk). These data suggest that, although Eirk Yale is weakly connected to the bank accounts inside the deepest cluster, he might play an important role in the fraudulent pattern, as well. According to the FinCEN's document description, this person was indeed the central actor of this illegal activity.

### 4.2. Qualitative evaluation

In order to get feedback about the usefulness of VisFAN in a real context, we prepared a questionnaire that was answered by 4 analysts of AIF, who are expert in the field of financial crime detection. This set of analysts includes the responsible for Organization and Administration of AIF. Note that, it is rather difficult to find out expert people in this field. The questionnaire was composed of 10 questions, which are reported below:

- *Q1* How much can VisFAN reduce the time and increase the productivity of the investigation processes in AIF?
- *Q2* How much can VisFAN increase the effectiveness (i.e., the ability of identifying potential criminal activities) of the investigation processes in AIF?

[3] http://www.fincen.gov/statutes_regs/files/sarnarrcompletguidfinal_112003.pdf.

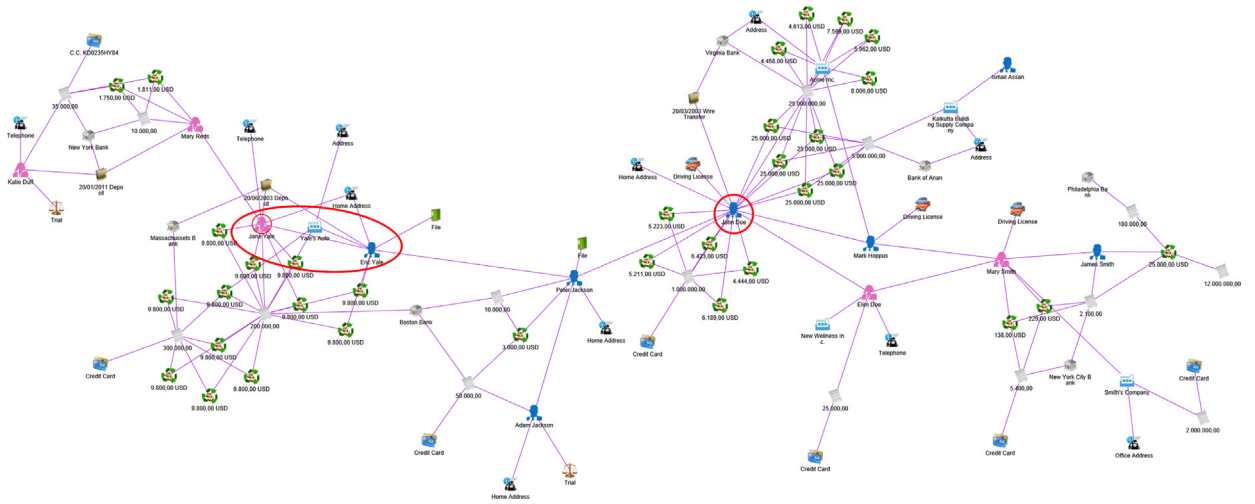

**Fig. 15.** A FAN containing fraudulent patterns. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

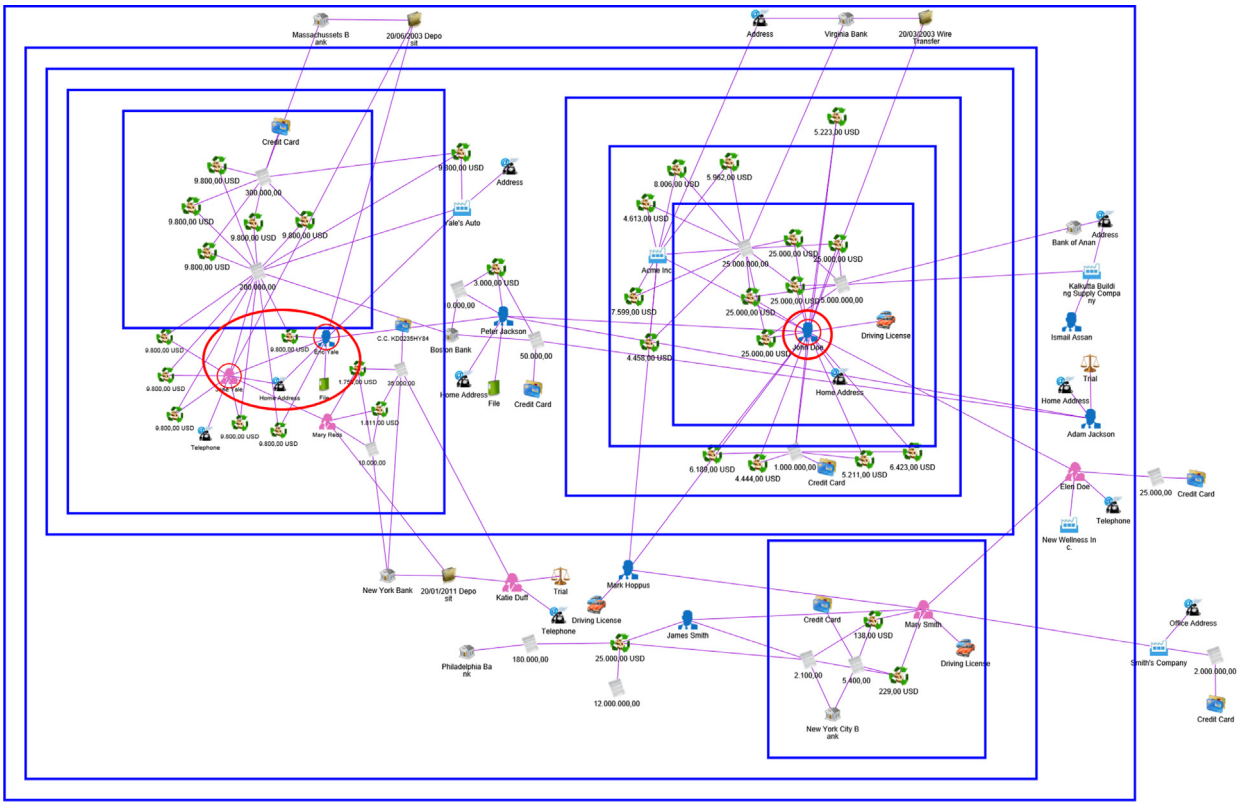

**Fig. 16.** The FAN of Fig. 15, after the execution of our clustering algorithm. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

*Q3* Within an investigation, how much do you benefit from the bottom-up exploration (incremental exploration) feature of VisFAN?

*Q4* Within an investigation, how much do you benefit from the top-down exploration (clustering) feature of VisFAN?

*Q5* How much do you benefit from the possibility of combining bottom-up and top-down exploration features of VisFAN in an effective way?

*Q6* Within an investigation, how much do you benefit from imposing the geometric constraints of the network layouts in VisFAN?

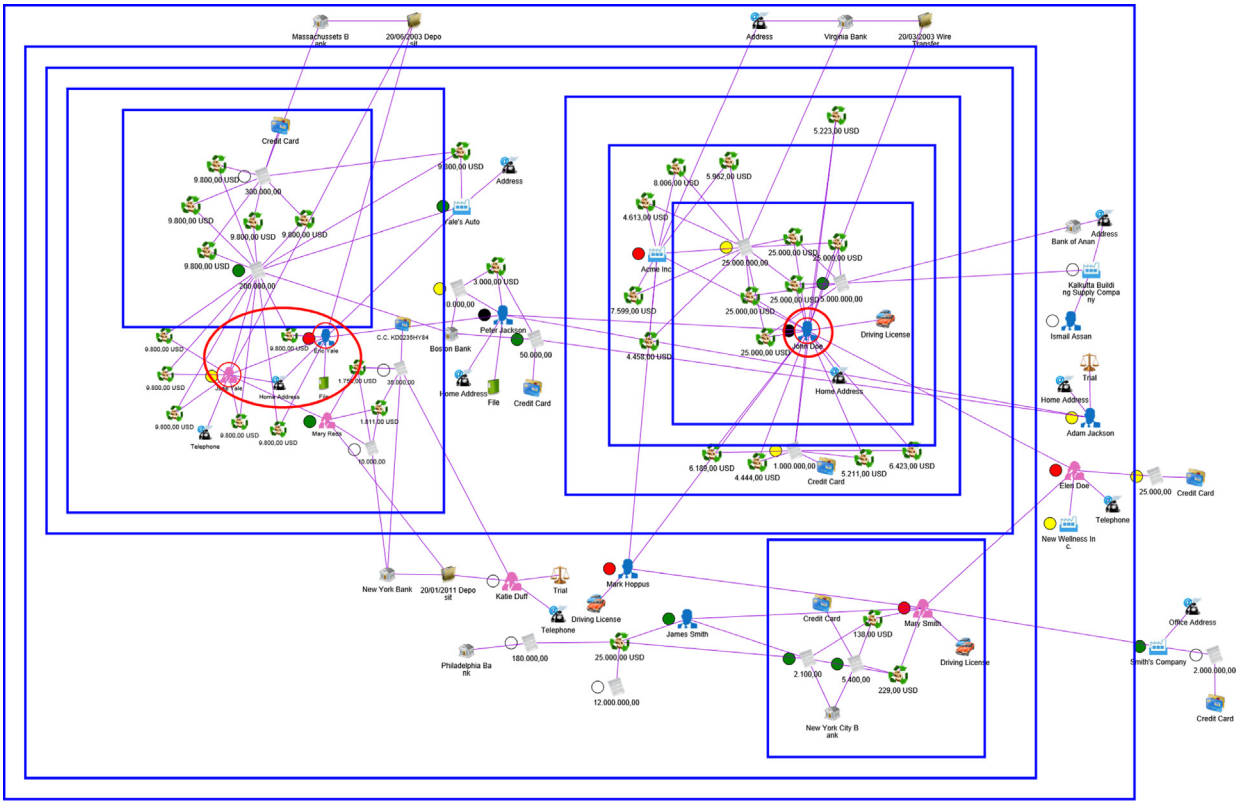


**Fig. 17.** The previous network where closeness node centralities are shown. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

*Q7* How much are you satisfied with the interaction level of VisFAN?
*Q8* Within an investigation, how much do you benefit from the set of centrality indexes of VisFAN?
*Q9* Within an investigation, how much do you benefit from the tree-based saving system of VisFAN?
*Q10* Within an investigation, how much do you benefit from the reporting functionality of VisFAN?

For each question the user gave a score in the set $\{1, 2, 3, 4\}$; the meanings of these scores are: 1: *nothing*; 2: *not much*; 3: *enough*; and 4: *a lot*. Table 1 reports the score assigned by each user to each question. Also the average score for each question is reported.

From the answers we can observe that all users perceive the system as a quite useful tool for their activity; in particular they judge more effective the use of each single interaction tool, like the incremental exploration, the clustering, and the geometric constraints (Q3, Q4, and Q6). They all agree that these tools help to speed-up the investigation process (Q1). Also the saving system is considered a key feature of the system (Q9). About question Q5, one user was not completely convinced by the combination of bottom-up and top-down exploration tools; we believe that this answer is due to the fact that combining these two functionalities in an effective way is not always straightforward, and as a consequence a longer learning time might be needed to let the analyst achieve a better experience with the system.

**Table 1**
Answers for the questionnaire.

| Question | User 1 | User 2 | User 3 | User 4 | Average |
|---|---|---|---|---|---|
| Q1 | 3 | 3 | 3 | 4 | *3.25* |
| Q2 | 3 | 3 | 3 | 3 | *3.0* |
| Q3 | 3 | 3 | 3 | 4 | *3.25* |
| Q4 | 4 | 3 | 3 | 3 | *3.25* |
| Q5 | 3 | 3 | 3 | 2 | *2.75* |
| Q6 | 3 | 3 | 3 | 4 | *3.25* |
| Q7 | 3 | 3 | 3 | 3 | *3.0* |
| Q8 | 3 | 3 | 3 | 3 | *3.0* |
| Q9 | 4 | 3 | 3 | 3 | *3.25* |
| Q10 | 3 | 3 | 3 | 3 | *3.0* |

## 5. Related work

The development of advanced methodologies and software systems for the analysis of criminal networks has received increasing attention after the September 11 terroristic attacks. A survey on these systems is presented by Xu and Chen [43]. The systems described in [43], and other newer systems, are briefly described in the following.

Chang et al. present the system WireVis, specifically tailored for the visual analysis of financial wire transactions

for fraud detection [11]. Tang et al. describe a social network analysis approach to detect illegal financial activities performed over the Internet [40]. Both the methodologies of Chang et al. and of Tang et al. use charts and plots, while they make little use of network visualization techniques.

Among the systems that use network visualization tools we mention I2 Analyst's NoteBook[4] [32], Netmap[5] [30], and Xanalysis Link Explorer[6] [3]. They implement classical layout algorithms to represent relational data, like force-directed algorithms, hierarchical layout algorithms, and circular drawing algorithms [14]. The research project COPLINK defines a system using hyperbolic trees to visualize large crime networks [13]. This system has been subsequently equipped with a classical spring-embedder algorithm and with some SNA tools, like association path searching and automatic clustering [12]. Another interesting system conceived to support investigation processes is Jigsaw [38]; it is mainly focused on extracting relevant entities (persons, addresses, dates, etc.) from a collection of documents, and to suggest possible correlations between these entities. It also provides a basic graph view displaying connections between entities and documents in a node-link diagram, allowing analysts to explore the documents by showing and hiding links and nodes.

Different from VisFAN, all systems listed above use basic implementations of classical graph drawing algorithms and lack of an effective integration of SNA tools within a highly interactive visualization environment. For example, they do not consider the possibility of visually interacting with clusters and/or having clustering techniques specifically designed for emphasizing fraudulent patterns by means of visual inspection.

VisFAN is built upon a prototype system, called VisForFraud [16], which handled only flat clustering and which allowed neither full combination of bottom-up and top-down interaction paradigms nor manual modification of the computed clusters. The drawing engine of VisFAN adopts a force-directed algorithm whose basic physical model is similar to that described by Fruchterman and Reingold [29]. However, in order to deal with clusters, layout constraints, and user interaction, the physical model has been engineered with extra elements and forces (see also the survey by Tamassia for the use of constraints in graph drawing [39]).

The top-down interaction paradigm used in VisFAN is inspired by that proposed by Eades and Huang [25]. However, different from [25], VisFAN makes it possible to modify the clustering during the exploration of the network and to enhance the network with new elements by automatically positioning them in the current cluster hierarchy. Also, the approach proposed by Eades and Huang does not deal with clusters of prescribed dimensions, while VisFAN allows the analyst to resize every single cluster region at her convenience. We observe that Frishman and Tal described a force-directed approach for dynamic clustered graphs, where vertices, edges, and clusters can be modified by the user [28]. However, the algorithm of Frishman and Tal is restricted to two-level clustering, and again the dimensions of each cluster is automatically computed and cannot be modified by the user. Dogrusoz et al. described a force-directed algorithm for drawing undirected compound graphs with nodes of different sizes [21]. However, their approach does not allow the user to interact with the drawing in order to enhance the network with new elements and to modify the cluster dimensions or to collapse clusters into single nodes.

Concerning the clustering process, do Nascimento and Eades proposed a semiautomatic framework, where the user can provide hints to the system [20]. Yet, their approach is restricted to two-level clustering, and it is aimed at computing a good partitioning of the vertices independent of their semantic. We also remark that several systems, other than that presented by Eades and Huang, adopt a top-down interaction paradigm based on clustering (see, e.g., [1,4,5,15,36,41]). Also, there exist systems that adopt a bottom-up exploration approach similar to that used by VisFAN (see, e.g., [10,24]). However, as far as we know, there is no previous system that allows for a full combination of the two paradigms.

We finally observe that VisFAN is able to compute several indexes of node centrality specifically defined for social networks and/or criminal networks, including those recently defined in [33,34]. The analyst can display the values of these indexes as node attributes in the layout. The importance of considering many different indexes of centrality in social graphs emerges from an experimental comparison performed by Le Merrer and Trédan [35].

## 6. Conclusions

The paper described a new software system for the analysis of financial activity networks; the aim is to support analysts in the discovery of criminal patterns, like money laundering and frauds. The main novelties of the system with respect to previous work in this field is the strong use of visual interactive tools, combined with ad-hoc clustering techniques and customizable layout constraints management. Furthermore, the system offers several additional facilities that help in the whole investigation process, and allows both the analysis of data within the same financial institution and the analysis of suspicious transaction data collected by FIUs from different financial subjects.

From the technology transfer point of view, the design of the system has been driven by the requirements of real-world FIU analysts, namely those working at the Financial Intelligence Agency (AIF) of the San Marino Republic. After the development of a first prototype, the system has been further engineered and become a commercial product. It is currently adopted by AIF, which gave us positive feedback about its usefulness, as reported in Section 4.2.

In the near future we plan to further evaluate the impact of our system on investigation processes. We will continue to cooperate with AIF for the next 2 years and enhance the system with additional functionalities, such as the capability of automatically extracting data from documents and the possibility of efficiently handling layouts of

[4] http://www.i2inc.com/.
[5] http://www.netmap.com/.
[6] http://www.xanalys.com/.

large networks by combining multi-scale force-directed algorithms with geometric constraints.

## Acknowledgments

We thank Dr. Giancarlo Montico and his staff at the Financial Intelligence Agency (AIF) of San Marino Republic for their valuable contribution and useful suggestions. We also thank all people that contributed in the development of VisFAN: Simona Amatucci, Benedetta Borella, Zamira Colacecchi and Michael Pinchi.

Finally, we thank the anonymous reviewers of this paper for their useful comments and suggestions.

## References

[1] J. Abello, F. van Ham, N. Krishnan, Ask-graphview: a large scale graph visualization system, IEEE Trans. Vis. Comput. Gr. 12 (5) (2006) 669–676.
[2] M. Altaf-Ul-Amin, K. Nishikata, T. Koma, T. Miyasato, Y. Shinbo, M. Arifuzzaman, C. Wada, M. Maeda, T. Oshima, H. Mori, S. Kanaya, Prediction of protein functions based on K-cores of protein–protein interaction networks and amino acid sequences, Genome Inf. 14 (2003) 498–499.
[3] T. Anderson, L. Arbetter, L. Benawides, A. Longmore-Etheridge, Security works, Secur. Manag. 38 (17) (1994) 17–20.
[4] D. Archambault, T. Munzner, D. Auber, Grouseflocks: steerable exploration of graph hierarchy space, IEEE Trans. Vis. Comput. Gr. 14 (4) (2008) 900–913.
[5] D. Auber, Y. Chiricota, F. Jourdan, G. Melançon, Multiscale visualization of small world networks, in: IEEE Information Visualization (INFOVIS 2003), 2003, pp. 75–81.
[6] V. Batagelj, F.-J. Brandenburg, W. Didimo, G. Liotta, P. Palladino, M. Patrignani, Visual analysis of large graphs using (X,Y)-clustering and hybrid visualizations, IEEE Trans. Vis. Comput. Gr. 17 (11) (2011) 1587–1598.
[7] V. Batagelj, A. Mrvar, M. Zaveršnik, Partitioning approach to visualization of large networks, in: Graph Drawing (GD 1999), Lecture Notes in Computer Science, vol. 1731, 1999, pp. 90–97.
[8] V. Batagelj, M. Zaversnik, An O(m) Algorithm for Cores Decomposition of Networks, CoRR, cs.DS/0310049, 2003.
[9] B. Bollobás, The evolution of sparse graphs, in: A. Press (Ed.), Graph Theory and Combinatorics, 1984, pp. 35–57.
[10] A. Carmignani, G. Di Battista, W. Didimo, F. Matera, M. Pizzonia, Visualization of the high level structure of the internet with HERMES, J. Gr. Algorithms Appl. 6 (3) (2002) 281–311.
[11] R. Chang, et al., Scalable and interactive visual analysis of financial wire transactions for fraud detection, Inf. Vis. 7 (1) (2008) 63–76.
[12] H. Chen, H. Atabakhsh, C. Tseng, B. Marshall, S. Kaza, S. Eggers, H. Gowda, A. Shah, T. Petersen, C. Violette, Visualization in law enforcement, in: ACM Conference on Human Factors in Computing Systems (CHI 2005), ACM, 2005, pp. 1268–1271.
[13] H. Chen, D.D. Zeng, H. Atabakhsh, W. Wyzga, J. Schroeder, Coplink: managing law enforcement data and knowledge, Commun. ACM 46 (1) (2003) 28–34.
[14] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis, Graph Drawing, Prentice Hall, Upper Saddle River, NJ, 1999.
[15] E. Di Giacomo, W. Didimo, L. Grilli, G. Liotta, Graph visualization techniques for web clustering engines, IEEE Trans. Vis. Comput. Gr. 13 (2) (2007) 294–304.
[16] E. Di Giacomo, W. Didimo, G. Liotta, P. Palladino, Visual analysis of financial crimes, in: Advanced Visual Interface (AVI 2010), ACM, 2010, pp. 393–394.
[17] W. Didimo, G. Liotta, Mining Graph Data, Chapter Graph Visualization and Data Mining, Wiley, Hoboken, New Jersey, 2007, 35–64.
[18] W. Didimo, F. Montecchiani, Fast layout computation of clustered networks: algorithmic advances and experimental analysis, Inf. Sci., 260 (2014) 185–199.
[19] W. Didimo, F. Montecchiani, Fast layout computation of hierarchically clustered networks: algorithmic advances and experimental analysis, in: Information Visualization (IV 2012), IEEE, 2012, pp. 18–23.
[20] H. A. do Nascimento, P. Eades, A system for graph clustering based on user hints, in: Visual Information Processing (VIP 2000), vol. 2 of CRPIT, ACS, 2001, pp. 73–74.
[21] U. Dogrusoz, E. Giral, A. Cetintas, A. Civril, E. Demir, A layout algorithm for undirected compound graphs, Inf. Sci. 179 (2009) 980–994.
[22] S.N. Dorogovtsev, A.V. Goltsev, J.F.F. Mendes, k-Core Organization of Complex Networks, CoRR, abs/cond-mat/0509102, 2005.
[23] P. Eades, A heuristic for graph drawing, in: Congressus Numerantium, vol. 42, 1984, pp. 149–160.
[24] P. Eades, R. F. Cohen, M. L. Huang, Online animated graph drawing for web navigation, in: Graph Drawing (GD 1997), vol. 1353, 1997, pp. 330–335.
[25] P. Eades, M.L. Huang, Navigating clustered graphs using force-directed methods, J. Gr. Algorithms Appl. 4 (3) (2000) 157–181.
[26] FinCEN—United States Department of the Treasury, Guidance on Preparing a Complete & Sufficient Suspicious Activity Report Narrative ⟨http://www.fincen.gov/⟩, 2003.
[27] L.C. Freeman, A set of measures of centrality based on betweenness, Sociometry 40 (1977) 35–41.
[28] Y. Frishman, A. Tal, Dynamic drawing of clustered graphs, in: IEEE Information Visualization, IEEE, 2004, pp. 191–198.
[29] T.M.J. Fruchterman, E.M. Reingold, Graph drawing by force-directed placement, Softw. Pract. Exp. 21 (11) (1991) 1129–1164.
[30] H.G. Goldberg, T.E. Senator, Restructuring databases for knowledge discovery by consolidation and link formation, in: Knowledge Discovery and Data Mining (KDD 1995), AAAI Press, 1995, pp. 136–141.
[31] A.V. Goltsev, S. N. Dorogovtsev, J. F. F. Mendes, k-core (bootstrap) percolation on complex networks, in: Critical Phenomena and Nonlocal Effects, CoRR, abs/cond-mat/0602611, 2006.
[32] P. Klerks, E. Smeets, The network paradigm applied to criminal organizations: theoretical nitpicking or a relevant doctrine for investigators? Recent developments in The Netherlands, Connections 24 (2001) 53–65.
[33] V. Latora, M. Marchiori, A measure of centrality based on the network efficiency, New J. Phys. 9 (2007).
[34] N. Memon, H.L. Larsen, Structural analysis and destabilizing terrorist networks, in: Data Mining (DMIN 2006), CSREA Press, 2006, pp. 296–302.
[35] E.L. Merrer, G. Trédan, Centralities: capturing the fuzzy notion of importance in social graphs, in: Social Network Systems (SNS 2009), ACM, 2009, pp. 33–38.
[36] D. Schaffer, Z. Zuo, S. Greenberg, L. Bartram, J. Dill, S. Dubs, M. Roseman, Navigating hierarchically clustered networks through fisheye and full-zoom methods, ACM Trans. Comput.-Human Interact. 3 (2) (1996) 162–188.
[37] S.B. Seidman, Network structure and minimum degree, Social Netw. 5 (1983) 269–287.
[38] J. Stasko, C. Görg, Z. Liu, Jigsaw: supporting investigative analysis through interactive visualization, Inf. Vis. 7 (April) (2008) 118–132.
[39] R. Tamassia, Constraints in graph drawing algorithms, Constraints 3 (April) (1998) 87–120.
[40] L. Tang, G. Barbier, H. Liu, J. Zhang, A social network analysis approach to detecting suspicious online financial activities, in: Advances in Social Computing, LNCS, 2010, pp. 390–397.
[41] F. van Ham, J. J. van Wijk, Interactive visualization of small world graphs, in: IEEE Information Visualization (INFOVIS 2004), 2004, pp. 199–206.
[42] C. Westphal, Data Mining for Intelligence, Fraud, & Criminal Detection, CRC Press, Boca Raton, FL, 2009.
[43] J. Xu, H. Chen, Criminal network analysis and visualization, Commun. ACM 48 (6) (2005) 101–107.