A Generalized Force-Directed Layout for Multiplex Sociograms

Zahra Fatemi[1], Mostafa Salehi[1], and Matteo Magnani[2(✉)]

[1] Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran
{fatemi.z,mostafa_salehi}@ut.ac.ir
[2] InfoLab, Department of Information Technology, Uppsala University, Uppsala, Sweden
matteo.magnani@it.uu.se

Abstract. Multiplex networks are defined by the presence of multiple edge types. As a consequence, it is hard to produce a single visualization of a network revealing both the structure of each edge type and their mutual relationships: multiple visualization strategies are possible, depending on how each edge type should influence the position of the nodes in the sociogram. In this paper we introduce *multiforce*, a force-directed layout for multiplex networks where both intra-layer and inter-layer relationships among nodes are used to compute node coordinates. Despite its simplicity, our algorithm can reproduce the main existing approaches to draw multiplex sociograms, and also supports a new intermediate type of layout. Our experiments on real data show that multiforce enables layered visualizations where each layer represents an edge type, nodes are well aligned across layers and the internal layout of each layer highlights the structure of the corresponding edge type.

Keywords: Visualization · Multiplex network · Layout
Force-directed

1 Introduction

Sociograms, that is, visual representations of the relationships between individuals, have been used since the origins of social network analysis, a notable example being Moreno's seminal work introducing this concept [22]. In the same book we also find a sociogram representing a multiplex network, where multiple types of relationships between the same group of individuals coexist. However, while several layout algorithms for simple graphs have been developed since then, developments in visualization methods for multiplex networks have been limited.

One natural way to visualize a multiplex network is to treat it as an edge-typed multi-graph using different colors and line styles to distinguish between the different edge types, as in Fig. 1(a) and as done by Moreno himself. However, this option can quickly lead to a very dense representation hiding relevant network structures even for very small networks [26]. Alternatively, different types of

S. Staab et al. (Eds.): SocInfo 2018, LNCS 11185, pp. 212–227, 2018.
https://doi.org/10.1007/978-3-030-01129-1_13

connections can be sliced into different layers, with the same node replicated on multiple layers, as in Fig. 1(b-d). This approach has been used in the literature to represent social/historical [21,23] networks, but also several other types of multiplex networks, from traffic [4,19] to biological [4] and financial [7] networks, sometimes visualized in a 2.5-dimensional space.

However, replicating the same node on multiple layers introduces a new question: how should the positions of multiple occurrences of the same node relate to each other? Two main approaches for visualizing multiplex networks sliced into layers have been used: visualizing each layer independently of the others, as in Fig. 1(b), or keeping the same layout in all layers, so that all occurrences of the same node will result aligned on a straight line if the layers are visualized one besides the other, as in Figs. 1(c) and (d).

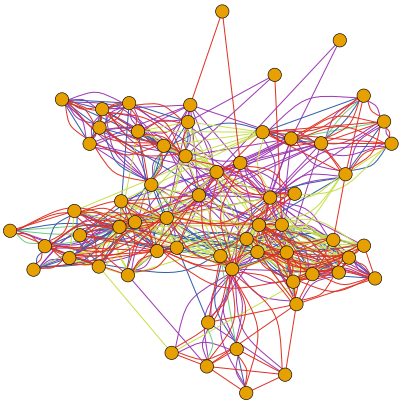
Using the first approach we can appreciate the structure of each layer. For example, in Fig. 1(b) we can see clear communities in the first (left-most) layer, representing the relationship *eating lunch together*, but less evident groups in the last layer about *working* relations. However, we get no information about the relationships between layers, e.g., whether the communities in the first layer correspond to tight groups also in the last.

Using the second approach it is easier to find the same nodes across different layers, but this does not necessarily help in understanding relationships between whole layers, and can also be misleading. As an example, in Fig. 1(c) all the nodes are aligned on the layout computed on the first layer, and this creates the illusion of the presence of similar communities in the fourth and fifth layers. Computing the layout on a combination of all edges in all layers or on one of the other layers, as in Fig. 1(d), has instead the effect of hiding the communities found in the first layer.

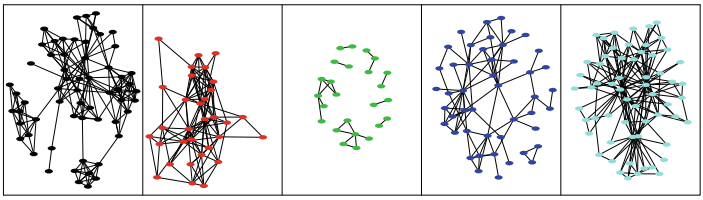
In this paper we claim that (1) the aforementioned approaches are just specific cases of a more general method, and can thus be produced by the same (simple) algorithm, and that (2) this algorithm can also produce new intermediate layouts between the ones mentioned above. These ideas are not new, as they were already explored in the context of dynamic graph drawing [8]. We call the general layout and algorithm described in this paper and tested on multiplex social networks *multiforce*.

Multiforce is based on a force-directed algorithm (in this paper we extend the popular Fruchterman-Reingold method) and uses two main types of forces: *intra-layer* and *inter-layer*, that can be tuned to impact specific layers more or less than others. Intra-layer forces attract neighbors inside the same layer, making them closer, as in traditional layouts for monoplex networks. Inter-layer forces try to align instances of the same node on different layers[1]. Figure 2 gives an intuition of how these forces operate. In addition, we use repulsive forces as in the original algorithm, and also gravity for the cases where no inter-layer forces are active and the network contains more than one component.

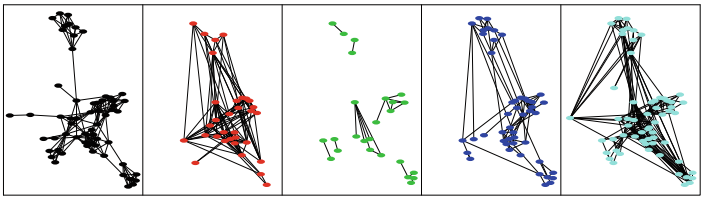
[1] In theory inter-layer forces can also be used to visualize more general networks, where edges can cross layers, but in this work we focus on multiplex networks.



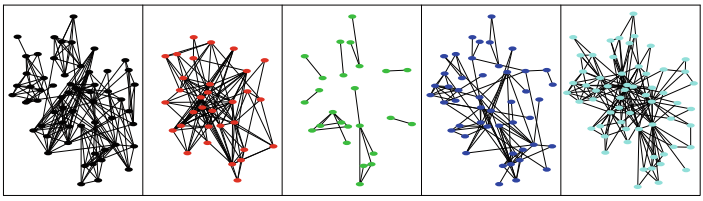
(a) Multi-graph (color figure). Communities are visible, edge types are mixed together



(b) Sliced, independent. Edge-type-specific communities are visible



(c) Sliced, aligned on first layer. Visible node correspondence across layers, phantom communities appear in other layers



(d) Sliced, aligned on last layer

Fig. 1. Four visualizations of the AUCS multiplex network

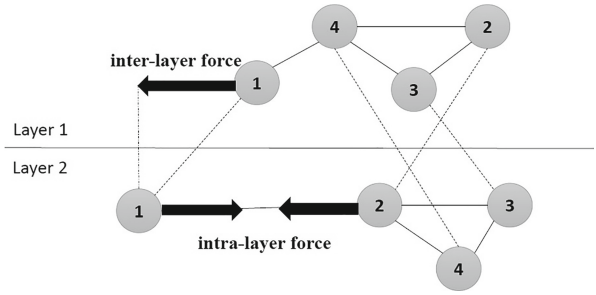


Fig. 2. The effect of intra-layer and inter-layer forces on node positions

To perform an objective comparison between our approach and a traditional non-multiplex force-based layout algorithm we have defined two quality metrics called respectively *external fit* and *internal fit*. These are naturally defined as the forces which would be active on the nodes if we were only trying to respectively align them across layers or draw them according to the internal structure of each layer. We have then used these metrics to characterize different settings of our general algorithm when applied to real social networks. In addition to this evaluation, we have also executed our algorithm on a simple dataset to qualitatively characterize the resulting diagrams.

2 The Multiforce Layout

Multiforce belongs to the slicing class of multiplex layouts, and is different from existing approaches, because it allows a balancing of the effects of intra-layer and inter-layer relationships. Multiforce extends the Fruchterman-Reingold algorithm [10]. This is one of the most popular options available in graph analysis software packages, although it is not a state-of-the-art method; several variations of this approach have been developed, but its simplicity allows us to focus on the simple variation of the way in which forces are defined, which can then be adapted to more complex algorithms. As mentioned in the introduction, multiforce is based on two types of attractive forces. The nodes are positioned on a set of planes, one for each layer or type of relationship – this setting is sometimes called 2.5-dimensional, because it looks 3-dimensional but the z-coordinates of the nodes are fixed and limited to the number of planes/layers. In this paper we visualize all layers on the same plane as in Fig. 1, because a 3-dimensional graph is visually intriguing but not easy to understand without the option of interactively rotating the diagram. Intra-layer forces, that can be weighted differently in each layer, attract pairs of nodes connected on the same layer. Inter-layer forces influence the position of nodes in different layers connected by inter-layer edges, or corresponding to the same node in the case of multiplex networks.

The same idea behind multiforce was already proposed and tested in the context of dynamic graphs [8], where layers are not unordered as in the case of multiplex networks but are organized in a sequence.

The pseudo-code of multiforce is presented in Algorithm 1, and the algorithm is implemented in the *multinet* library[2]. The algorithm takes a multiplex network $G = (N, L, V, E)$ as input, where N is a set of nodes, L is a set of layers, (V, E) is a graph and the elements of V are pairs ⟨node, layer⟩. We notate v.layer the layer of an element $v \in V$ and v.node the node corresponding to an element $v \in V$.

Lines 13–29 are the same as in the original algorithm, and compute the displacement of each node based on its neighbors (attractive forces) and other nodes (repulsive forces), with the addition of weights that can be used to specify on which layers the layout should be computed according to the original algorithm (27–28). Lines 30–37 extend the original algorithm and compute the displacement caused by the position of the node on other layers, to control node alignment. This is also weighted, to allow the user to turn this feature on and off for all or some layers (34–35). In our tests the function *cool* (45) reduces t linearly, so that it becomes 0 at the last iteration.

Some details of our algorithm can also be changed without affecting its underlying idea. First, we can modify lines 6–12 to assign the same initial random coordinates to the same node across different layers, anticipating line 8 before the for loop. A weighting factor INLA[v] can also be added at line 20, so that both attractive and repulsive forces are reduced or reinforced together. Finally, lines 41 and 42 have been retained from the original algorithm and ensure that the nodes do not exit the frame specified by the user, but are not necessary if the final coordinates are re-scaled to fit it or a gravity force is added to control the spreading of the nodes so that all slices retain similar extreme coordinates. In this paper gravity is only used in the examples with no inter-layer forces and multiple components on the same layer.

2.1 Main Settings

The multiforce algorithm can produce both existing and new layouts using the following settings:

1. **Multi-graph:** this layout, where each node has a specific position that does not depend on the layer and all edge types are considered when computing the node coordinates, is obtained by setting the same positive value for intra-layer weight in each layer and infinite (or, in practice, very high) inter-layer weights. The intra-layer weights will then keep nodes aligned across layers, and intra-layer forces will produce the layout by moving these "node pillars" around.
2. **Sliced, independent**: this layout corresponds to the application of the force-based algorithm on each layer, and is obtained by using positive intra-layer weights and setting inter-layer weights to 0.
3. **Sliced, aligned on layer x:** this layout is computed based on one of the layers, and nodes are kept aligned on the other layers. It is obtained by

[2] https://cran.r-project.org/package=multinet.

Algorithm 1. Multiforce

Require: $G = (N, L, V, E)$: a multiplex network
Require: W: width of the frame
Require: L: length of the frame
Require: #iterations
Require: INLA, INTERLA: intra- and inter-layer weights
f_r = function(z, k)\{ return k^2/z; \}
f_a = function(z, k)\{ return z^2/k; \}
area := $W \cdot L$
$k := \sqrt{\frac{\text{area}}{|N|}}$;
$t := \sqrt{|N|}$;
for $(n \in N)$ **do**
  for $(v \in V$ s.t. v.node $= n)$ **do**
    (x, y) = random coordinates;
    pos$[v]$ = (x, y)
    $z[v]$:= index(v.layer);
  end for
end for
for ($i = 1$ to #iterations) **do**
  // calculate repulsive forces
  for $(v \in V)$ **do**
    disp$[v]$:= **0**;
    for $(u \in V)$ **do**
      if $(u \neq v$ and u.layer $= v$.layer) **then**
        $\triangle$:= pos$[v]$ − pos$[u]$;
        disp$[v]$:= disp$[v]$ + $(\triangle/|\triangle|) * f_r(|\triangle|)$;
      end if
    end for
  end for
  // calculate attractive forces inside each layer
  for $((u, v) \in E)$ **do**
    $\triangle$:= pos$[v]$ − pos$[u]$;
    disp$[v]$:= disp$[v]$ − $(\triangle/|\triangle|) * f_a(|\triangle|, k) *$ INLA$[v]$;
    disp$[u]$:= disp$[u]$ + $(\triangle/|\triangle|) * f_a(|\triangle|, k) *$ INLA$[u]$;
  end for
  // calculate attractive forces across layers
  for $(n \in N)$ **do**
    for $(\{u, v\}$ with $u, v \in V$, u.node $= v$.node $= n)$ **do**
      $\triangle$:= pos$[v]$ − pos$[u]$;
      disp$[v]$:= disp$[v]$ − $(\triangle/|\triangle|) * f_a(|\triangle|, k) *$ INTERLA$[v, u]$;
      disp$[u]$:= disp$[u]$ + $(\triangle/|\triangle|) * f_a(|\triangle|, k) *$ INTERLA$[u, v]$;
    end for
  end for
  // assign new positions
  for $(v \in V)$ **do**
    pos$[v]$:= pos$[v]$ + (disp$[v]$/|disp$[v]$|) $*$ min(disp$[v]$, t);
    pos$[v].x$:= $\min(W/2, \max(-W/2, \text{pos}[v].x))$;
    pos$[v].y$:= $\min(L/2, \max(-L/2, \text{pos}[v].y))$;
  end for
  // reduce the temperature
  t := cool(t);
end for

specifying a positive intra-layer weight for layer x and setting the other intra-layer weights and the inter-layer weights to 0.
4. **Balanced:** this intermediate layout is obtained by setting a positive weight (for example, 1) for all inter- and intra-layer forces.

In Sect. 3 we experimentally study the stability of the layouts depending on the chosen weights, showing that they are very stable and largely independent of the specific values we use for the positive weights.

2.2 Time Complexity

When the number of layers is constant, the asymptotic time complexity of multiforce is the same as for simple networks, with a larger constant. Separating nodes based on repulsive forces requires to compare each of the $|N|$ nodes on each layer with all the other nodes on the same layer, for a total time complexity of $O(|N|^2|L|)$ if no spatial indexes are used. Computing inter-layer forces on a node requires access to all its replicas, one on each other layer, for a total complexity of $O(|L|^2|N|)$. For a complete network with $|N|$ nodes and $|L|$ layers, there are at most $(|L|\frac{|N|(|N|-1)}{2})$ intra-layer edges contributing to attractive forces. In the typical case when $|L|$ is small or significantly smaller than $|N|$ the time complexity of drawing a multiplex network with the multiforce layout is thus $O(|N|^2)$ assuming a constant number of iterations. In general, the complexity is $O(|N|^2|L| + |L|^2|N|)$.

3 Experimental Evaluation

In this section we present an experimental evaluation of the new type of layout (balanced) enabled by multiforce, divided into two parts. First, we provide a quantitative evaluation where we show how the algorithm can produce layouts that are close to an ideal case when executed on a number of real datasets. Then, we execute the algorithm on the real dataset shown at the beginning of the article, to provide an additional qualitative analysis.

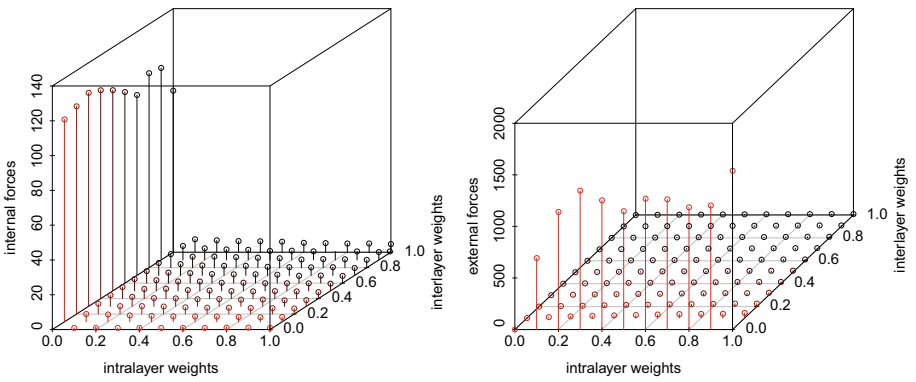
Please notice that we are not claiming that a balanced layout is the best possible: each of the layouts we can obtain using multiforce, including the ones already existing in the literature, highlights some aspects of the network. Therefore, it is important to be able to produce many of them during an exploratory analysis, and to be able to choose among them when presenting an analysis depending on what aspect we want to emphasize. The additional focus we put on the balanced layout in this section is only due to the fact that the other layouts that can be produced by multiforce have already been used and described in previous works.

3.1 Quantitative Evaluation

In this section we compare different layouts against an ideal case where each layer is drawn independently of the others, to optimize its internal readability,

and at the same time all occurrences of the same node across different layers are kept perfectly aligned. Notice that such an ideal visualization is impossible to obtain in general, except when the two layers are (almost) identical. To compute the distance between the ideal case and our tests we can define two measures representing these two criteria: the sum of the forces still active on the nodes inside each layer (called internal fit) and the displacement between nodes in different layers (called external fit), also computed as the sum of the inter-layer forces still active at the end of the algorithm. Notice that an optimal diagram as defined above would contain both internally stable and aligned nodes, so both measures would be close to 0, indicating the best possible layout (assuming that a force-based approach is used).

In this context, we perform two experiments. The first tests the effect of the parameters (intra- and inter-layer forces) on the external and internal fit when applied to our working example. Figure 3 shows that the results are very stable with respect to the input parameters. We remind the reader that the objective is to achieve low values for both internal and external fit.



(a) internal fit (b) external fit

Fig. 3. Effect of parameters on internal and external fit. Low internal and external forces (vertical axis) correspond to an ideal layout. For most combinations of inter- and intra-layer weights both forces are low.

In addition to the indication of the stability of the parameters, Fig. 3 suggests that a balanced layout can obtain good scores on both internal and external fit at the same time when using both intra-layer and inter-layer forces. We compare the results of the various approaches on the real datasets summarized in Table 1:

- A hybrid online/offline social network with five types of relationships among the employees of a CS department [26], also used as our working example.
- Traditional multiplex networks from the Social Network Analysis literature.
- A dataset showing the relationship among physicians in four different cities, collected to investigate information diffusion about drugs [13].

– Two transport networks, one with flight connections in Europe [3] and one about the London underground [5].

Table 1. Properties of real-world networks

Network	# Layers	# Nodes	# Edges
Social/Hybrid – AUCS	5	61	1240
Social/Historical – Padgett	2	15	35
Social – Krackhardt High Tech	3	16	166
Social – Lazega Law Firm	3	212	1663
Social – Physicians	3	241	1370
Transport – EU airlines	37	417	3588
Transport – London	3	369	441

Table 2. Results of experiments on real data

Network	Balanced		Independent		Aligned (avg)		Multi-graph	
	Inter	Ext	Inter	Ext	Inter	Ext	Inter	Ext
AUCS	4.78	9.17	1.03	1489.21	149.72	0.15	112.713	0.02
Padgett-Florentine-Families	2.02	3.71	0.27	67.19	26.53	0.07	10.899	0.00
Krackhardt High Tech	4.92	9.79	0.40	84.86	117.02	0.06	19.766	0.00
Lazega Law Firm	4.61	8.40	1.96	40.26	55.73	0.05	551.532	0.02
Physicians	5.06	6.36	2.93	1342.36	66.83	0.46	1451.43	0.10
EU Air	3.66	6.12	1.58	3142.33	217.67	0.30	851.04	0.98
London	2.73	1.52	2.10	138.84	95.46	0.04	131.88	0.03

For each network, Table 2 shows how different layouts behave with respect to the two aforementioned measures (internal fit and external fit) in the plots. We remind the reader that high values of the former means that some layers have not been correctly visualized according to their internal structure, for example, the nodes in a community may have been spread around instead of being close to each other. A high value of the latter means that nodes are not aligned across layers, for example, a node may have been visualized in the top left corner in one layer and in the top right in the other. The ideal case would be to have 0 for both metrics. For each network, several settings have been tested.

As expected, for each network the independent visualization generates nodes that are not well aligned, represented by high external forces active on the nodes. On the contrary, computing the layout based on one layer prevents other layers from having good internal layouts, as shown by the high internal forces active

on the nodes when the aligned strategy is used. The balanced option (first case) presents internal layouts that are less good than the ideal case, and node alignments that are also less good than the best possible option, but both are significantly better than the aspect not optimized using other strategies. In practice, this corresponds to layouts similar to the one shown in Fig. 4. The multi-graph approach presents results that in many cases may seem similar to the average of the results obtained using the aligned approach. However, consider that the results for the aligned approach are an average, where for each test the layout is computed on a different layer. Therefore, with the aligned approach there is always one layer with nearly optimal internal fit (that is, with a very low value).

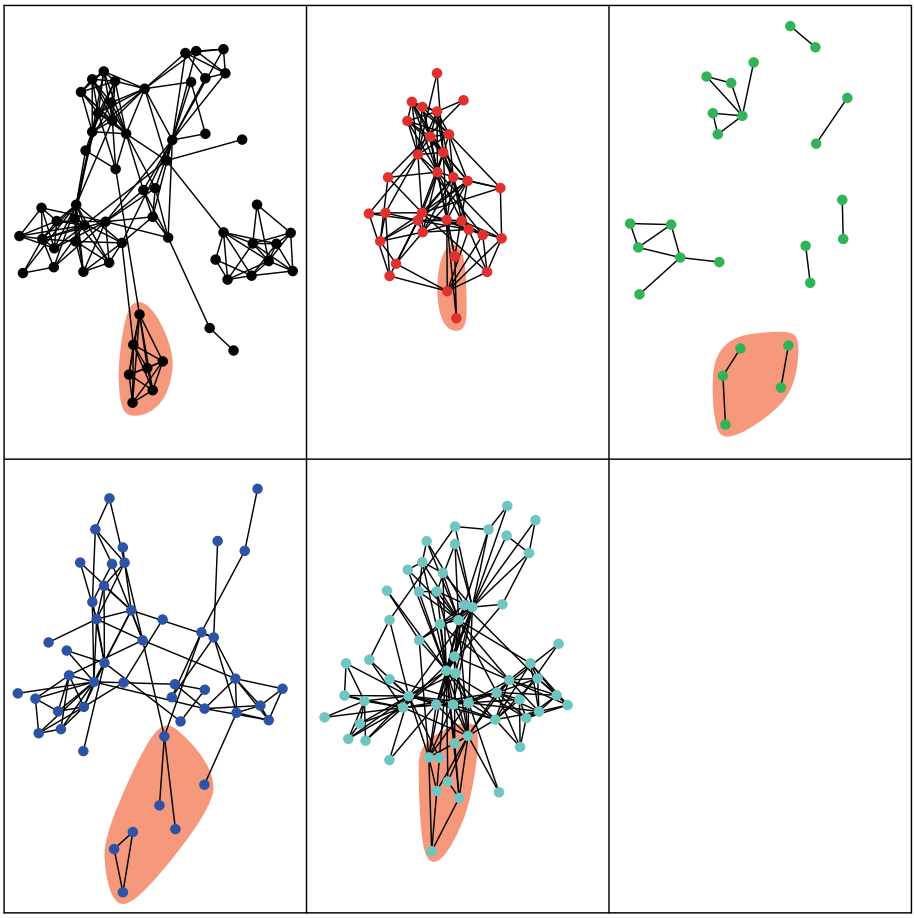


Fig. 4. Alignment of network structures across layers using a balanced visualization, with both intra-layer and inter-layer forces active. One community in the first layer has been highlighted, and the same nodes are also highlighted on the other layers (when present)

3.2 Qualitative Evaluation of the New Balanced Layout

In Fig. 4 we have visualized the 5-layer social network already shown in the introduction using both inter- and intra-layer forces, following the intuition discussed in the previous section about the ability of this setting to keep nodes reasonably aligned while preserving intra-layer structures. Each layer has been drawn according to their internal organization, showing peculiar structures: for example, the nodes in the community marked in the top-left diagram are not present in all the other layers, and where present they are not so well connected to each other. However, we can also see how the nodes belonging to this community have been visualized at similar locations in the other layers, providing information about inter-layer relationships. We can see, for example, that the community is split into two sub-communities in the third (top-right) layer, that it is not present in the fourth layer, where the same nodes are connected to different parts of the graph but a core subset of them is still forming a triangle, and that they form a similar but less dense community in the fifth layer. Notice that in the fourth layer the nodes are still more or less in the same position as in the first layer, despite the fact that they do not form a community. In this way it becomes easier to locate them – in an interactive plot we can also use a 2.5-dimensional visualization as in the previous section, making the task straightforward as long as the nodes are more or less aligned.

4 Related Work

In this section we describe existing approaches to compute network layouts with a single type of edges, also known as monoplex networks, and for multiplex networks.

4.1 Monoplex Network Visualization

Many layouts have been designed to visualize monoplex networks. Here we briefly review the ones that are more relevant for our approach. For an extensive review, the reader may consult [29].

Multi-scale layouts first create some core sub graphs, then they add other nodes until all nodes and edges have been processed [17,29]. *Random layouts* [6] and *circular layouts* [20] are two categories of layouts which are appropriate for small graphs with few nodes and edges, because they do not consider aesthetic criteria: many edge crossings and node overlappings can appear.

Among the most used visualization methods, *force-directed* algorithms consider a graph as a physical system where forces change the position of nodes. The two best known force-directed layouts are *Fruchterman-Reingold* [10] and *Kamada-Kawai* [14,15]. In the Fruchterman-Reingold layout nodes have repulsive power and push other nodes away, while edges attract neighboring nodes. In this layout nodes are considered as steel rings having similar loads and edges are like springs attracting neighbouring rings. This algorithm consists of three main

steps. First, all nodes are distributed randomly. Second, repulsive forces separate nodes. The value of repulsive force depends on the positions of the nodes. Third, for each edge and based on the position of nodes after repulsion, attractive forces are calculated [10]. In the Kamada-Kawai layout an energy function is defined for the whole graph based on shortest paths between nodes, and positions are iteratively updated until the graph's energy is minimized [15].

Bannister et al. [1] proposed a force-directed layout to change the position of nodes so that more graph-theoretically central nodes are pushed towards the centre of the diagram. In this algorithm, an additional force called gravity is used to change the position of more central nodes. For each node v in a graph G the position of the node is influenced by the following force:

$$I[v] = \sum_{u,v \in V} f_r(u,v) + \sum_{(u,v) \in E} f_a(u,v) + \sum_{v \in V} f_g(v) \tag{1}$$

where f_r and f_a are respectively repulsive and attractive forces, and f_g is the gravity force, measured as:

$$f_g(v) = \gamma_t M[v](\xi - P[v]) \tag{2}$$

In this equation $M[v]$ is the mass of node v, which can be set according to node degree, $P[v]$ is the position of v, γ_t is the gravitational parameter and $\xi = \Sigma_v P[v]/|V|$ is the centroid of all nodes. Notice that forces in the equations above are vectors.

Other extensions of force-directed layouts have considered the inclusion of additional domain-specific information in the definition of the forces. An example is [34], where the definitions of attractive and repulsive forces include terms representing trust in social networks.

Traditional force-directed methods are suitable only for small networks, but they are very popular because they often practically succeed in highlighting communities (that is, groups of well connected nodes) and increasing graph readability. To address their shortcomings, force-based layout algorithms are sometimes split into multiple phases, with an initial preprocessing of the data to generate good starting node dispositions or to reduce data complexity. As an example, Gajer et al.'s method first partitions the graph into subgraphs [31]. The smallest subgraph is then processed independently and thus more efficiently. Afterwards, a force-directed refinement round changes the values of initial node positions and the next smallest subgraph is added to the previous one, with these steps being repeated untill all nodes have been processed. [35] follows the same steps, setting initial node positions in a different way. Similarly, Peng et al. proposed the Social Network Analysis Layout (SNAL) by also separating a network into subgroups, analyzing relationships between them and positioning nodes based on their relative centralities [32]. Recent works have shown how to use force-based approaches on relatively large graphs [33].

Another family of layouts, that can also be combined with force-directed algorithms, are *constraint-based* layouts [9]. These layouts force nodes to appear at specific positions. For example, nodes are placed on a frame in a way that they

do not overlap, or are horizontally and vertically aligned, as in the *orthogonal* layout [9,29]. In these layouts it is more difficult to isolate special structures such as communities and time complexity is noticeably high.

One important assumption in graph drawing is a correspondence between some aesthetic features of the diagrams and their readability. Therefore, some visualization algorithms explicitly target these features. One such criterion is that too many edge crossings make a graph more difficult to interpret. The crossing number $cr(G)$ of a graph G is the smallest number of crossings appearing in any drawing of G [27]. Several algorithms have been proposed to reduce edge crossing in monoplex networks. For example, Shabbeer et al. [28] developed a stress majorization algorithm, as initial proposed in [11]. In [27] and [2] the concept of edge crossing is elaborated and equations for measuring the number of edge crossings in different graphs are reviewed. Another aesthetic feature impacting graph readability is node overlapping. Two popular methods to reduce node overlapping were proposed in [12,18].

4.2 Multiplex Network Visualization

Different methods have been discussed for visualizing multiplex networks. We categorize these methods into four main classes: slicing, flattening, simplification-based, and indirect.

One way of visualizing multiplex networks is to consider each layer or relationship type as a monoplex network and to connect these monoplex networks using inter-layer edges [4]. The layers can have aligned layouts or independent layouts, as shown in our initial example. Aligned layouts help users find similar nodes in different layers by forcing the same node to have the same coordinates on all layers, but structures existing only on one layer (for example communities) may not be clearly visualized. Independent layouts can show specific structures of each layer, but may hide inter-layer patterns [26].

In flattening-based methods all nodes and edges are placed on the same plane. In a *node-colored* network, nodes from different layers are shown with different colors [16], while for multiplex networks colors can be used to distinguish edges of different types. Apart from suffering from the same problems of aligned slicing, the disadvantage of this method is that for networks with high edge density relationships among nodes can be hidden by edges from non-relevant layers and readability quickly decreases due to the network's clutter.

Given the information overload associated to the presence of multiple types of edges, a third approach consists in adding a so called *simplification* step before using one of the two approaches above [26]. The simplification step removes edges or layers that are not considered relevant for the visualization task. In general, this can also be used as a pre-processing step before computing our multiforce layout.

The last approach tries to visualize information derived from the network instead of directly visualizing the original layers, which are again considered to be too complex to allow a simple visual representation. Renoust et al. [25] proposed a system for visual analysis of group cohesion in flattened multiplex

networks. This system, called *Detangler*, creates a so-called substrate network from unique nodes of the multiplex network and a so-called catalyst network from edges of different types. Erten et al. [30] proposed three modified force-directed approaches for creating slicing, flattened and split views of multilayer networks by considering edge weights and node weights. The weights of nodes and edges are based on the number of times they appear in different layers. In this approach interlayer relationships between nodes are ignored and node weights are the same for all nodes when multiplex networks are visualized, so this approach does not consider the specific features of the networks targeted in our work.

An extreme case of indirect methods, that we mention for completeness, consists in not visualizing nodes and edges at all but only indirect network properties, such as the degree of the nodes in the different layers or other summary measures [4,24,26]. These approaches are complementary to graph drawing, and can also be used in combination with our proposal.

5 Conclusion

A theoretically optimal force-based layout for multiplex networks would be able to reveal the structure of each layer and the relationships between different layers at the same time. Unfortunately, this is not possible in general, because these two aspects may not be aligned in real data, with some layers not being similar to the others.

To address this problem, we proposed multiforce, a force-directed algorithm in which both intra-layer and inter-layer forces can affect the position of nodes. Intra-layer forces keep connected nodes together and improve the identification of communities, while inter-layer forces help users finding the same nodes in different layers.

In the evaluation of the method we showed that while the algorithm supports more traditional layouts it can also generate what we call *balanced visualizations* where both internal properties and node alignments are handled. This has been presented on a real dataset, to give a qualitative idea of how the layouts produced by this approach look like, and quantitatively, introducing two quality metrics and showing how a balanced approach can satisfactorily address both at the same time on several real datasets from different domains. At the same time, traditional approaches are better at optimizing either one or the other quality metric. We thus believe that the main value of multiforce is its ability to represent all the strategies mentioned in the article, so that an analyst can switch from one to the other depending on which aspects s/he wants to focus on.

Acknowledgements. We thank Prof. Ken Wakita for his comments on an early version of this manuscript, and Prof. Mats Lind for insightful discussions. The work by Matteo Magnani has been funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 732027.

References

1. Bannister, M.J., Eppstein, D., Goodrich, M.T., Trott, L.: Force-directed graph drawing using social gravity and scaling. In: Didimo, W., Patrignani, M. (eds.) GD 2012. LNCS, vol. 7704, pp. 414–425. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36763-2_37
2. Buchheim, C., Chimani, M., Gutwenger, C., Jnger, M., Mutzel, P.: Crossings and Planarization. CRC Press, Hoboken (2013)
3. Cardillo, A., Gómez-Gardeñes, J., Zanin, M., Romance, M., Papo, D., del Pozo, F., Boccaletti, S.: Emergence of network features from multiplexity. Sci. Rep. **3**, 1344 (2013)
4. De Domenico, M., Porter, M.A., Arenas, A.: MuxViz: a tool for multilayer analysis and visualization of networks. J. Complex Netw. **3**, 159–176 (2014)
5. De Domenico, M., Solé-Ribalta, A., Gòmez, S., Arenas, A.: Navigability of interconnected networks under random failures. PNAS **111**, 8351–8356 (2014)
6. Díaz, J., Petit, J., Serna, M.: A survey of graph layout problems. ACM Comput. Surv. **34**(3), 313–356 (2002)
7. Dwyer, T., Gallagher, D.R.: Visualising changes in fund manager holdings in two and a half-dimensions. Inf. Vis. **4**(3), 227–244 (2004)
8. Brandes, U., Mader, M.: A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In: van Kreveld, M., Speckmann, B. (eds.) GD 2011. LNCS, vol. 7034, pp. 99–110. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-25878-7_11
9. Dwyer, T., Marriott, K., Schreiber, F., Stuckey, P., Woodward, M., Wybrow, M.: Exploration of networks using overview+detail with constraint-based cooperative layout. IEEE Trans. Vis. Comput. Graph. **14**(6), 1293–300 (2008)
10. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Softw. Pract. Exp. **21**(11), 1129–1164 (1991)
11. Gansner, E.R., Koren, Y., North, S.: Graph drawing by stress majorization. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 239–250. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31843-9_25
12. Huang, X., Lai, W., Sajeev, A.S.M., Gao, J.: A new algorithm for removing node overlapping in graph visualization. Inf. Sci. **177**(14), 2821–2844 (2007)
13. Coleman, H.M.J., Katz, E.: The diffusion of an innovation among physicians. Sociometry **20**(4), 253–270 (1957)
14. Moody, J., McFarland, D., Bender deMoll, S.: Dynamic network visualization. Am. J. Sociol. **110**(4), 1206–1241 (2005)
15. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. Inf. Process. Lett. **31**(1), 7–15 (1989)
16. Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J.P., Moreno, Y., Porter, M.A.: Multilayer networks. J. Complex Netw. **2**(3), 203–271 (2014)
17. Koren, Y., Carmel, L., Harel, D.: ACE: a fast multiscale eigenvectors computation for drawing huge graphs. In: IEEE Symposium on Information Visualization, INFOVIS 2002, pp. 137–144 (2002)
18. Kumar, P., Zhang, K.: Visualization of clustered directed acyclic graphs with node interleaving. In: Proceedings of the 2009 ACM Symposium on Applied Computing, SAC 2009, New York, NY, USA, pp. 1800–1805. ACM (2009)
19. Kurant, M., Thiran, P.: Layered complex networks. Phys. Rev. Lett. **96**(13), 138701 (2006)

20. Ma, D.: Visualization of social media data: mapping changing social networks. Master's thesis, the Faculty of Geo-Information Science and Earth Observation of the University of Twent (2012)
21. Magnani, M., Rossi, L.: The ML-model for multi-layer social networks. In: Proceedings of the International Conference on Social Network Analysis and Mining (ASONAM), pp. 5–12. IEEE Computer Society (2011)
22. Moreno, J.: Who Shall Survive?: A New Approach to the Problem of Human Interrelations. Nervous and Mental Disease Publishing Co., Washington, D.C. (1934)
23. Padgett, J.F., McLean, P.D.: Organizational invention and elite transformation: the birth of partnership systems in renaissance florence. Am. J. Sociol. **111**(5), 1463–1568 (2006)
24. Redondo, D., Sallaberry, A., Ienco, D., Zaidi, F., Poncelet, P.: Layer-centered approach for multigraphs visualization. In: Proceedings of the International Conference on Information Visualisation (iV), pp. 50–55 (2015)
25. Renoust, B., Melanon, G., Munzner, T.: Detangler: visual analytics for multiplex networks. Comput. Graph. Forum **34**(3), 321–330 (2015)
26. Rossi, L., Magnani, M.: Towards effective visual analytics on multiplex and multilayer networks. Chaos Solitons Fractals **72**, 68–76 (2015)
27. Schaefer, M.: The graph crossing number and its variants: a survey. Electron. J. Comb. **1000**, DS21 (2013)
28. Shabbeer, A., Ozcaglar, C., Gonzalez, M., Bennett, KP.: Optimal embedding of heterogeneous graph data with edge crossing constraints. In: Presented at NIPS Workshop on Challenges of Data Visualization, p. 1 (2010)
29. von Landesberger, T., et al.: Visual analysis of large graphs: state-of-the-art and future research challenges. Comput. Graph. Forum **30**(6), 1719–1749 (2011)
30. Erten, C., Kobourov, S.G., Le, V., Navabi, A.: Simultaneous graph drawing: layout algorithms and visualization schemes. In: Liotta, G. (ed.) GD 2003. LNCS, vol. 2912, pp. 437–449. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24595-7_41
31. Gajer, P., Goodrich, M.T., Kobourov, S.G.: A multi-dimensional approach to force-directed layouts of large graphs. Comput. Geom. **29**(1), 3–18 (2004)
32. Peng, W., SiKun, L.: Social network analysis layout algorithm under ontology model. SOFTWARE **6**(7), 3–18 (2011)
33. Ortmann, M., Klimenta, M., Brandes, U.: A sparse stress model. J. Graph Algorithms Appl. **21**(5), 791–821 (2017)
34. Ma, N., Lu, Y., Gan, H., Li, Z.: 2013 10th Web Information System and Application Conference - Trust Network Visualization Based on Force-Directed Layout, (wisa), yangzhou, china, 10 November 2013–15 November 2013 (2013)
35. Baur, M., Brandes, U., Gaertler, M., Wagner, D.: Drawing the AS graph in 2.5 dimensions. In: Pach, J. (ed.) GD 2004. LNCS, vol. 3383, pp. 43–48. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31843-9_6