Computer Science and Information Technology 1(3): 225-232, 2013
DOI: 10.13189/csit.2013.010308 http://www.hrpub.org

A Graph-based Overview Visualization for Data Landscapes

Grigor Tshagharyan[1], **Hans-Jörg Schulz**[2,*]

[1]Faculty of Informatics and Applied Mathematics, Yerevan State University, 0025 Yerevan, Armenia
[2]Faculty of Computer Science and Electrical Engineering, University of Rostock, 18051 Rostock, Germany
*Corresponding Author: hjschulz@informatik.uni-rostock.de

Abstract In many domains, it becomes more and more common that an analysis spans various interlinked data sources that we collectively term data landscape. Yet for the selection of appropriate data sources from the wide range of available ones, current approaches and systems rarely offer more support than a File-Open-dialog. This paper presents a visualization approach that aims to give a stable and meaningful overview of a data landscape to ease finding and selecting data sources that may be useful in a subsequent visual analysis. As such, it serves as a visual starting point from which to bootstrap a visual analysis by finding and selecting the data sources relevant to a question at hand. This approach is exemplified by applying it to a current snapshot of the data landscape from the CKAN-LOD data hub consisting of 216 data sources with 613 links between them.

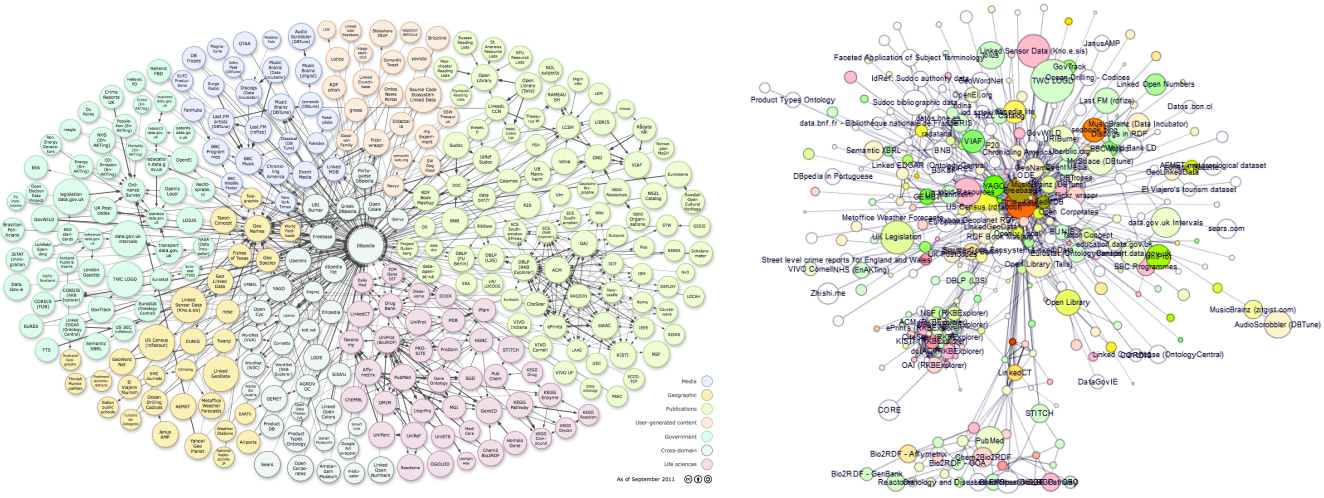
Keywords Linked Open Data, LOD Cloud, Multimodal Data Visualization

1 Introduction

Data sources are increasingly interlinked as it becomes more frequent that datasets are no longer self-contained, but reference other data sources via common identifiers. As a result, a typical visual analysis is rarely confined to a single data source, but spans across multiple such interlinked data sources to yield meaningful and reliable insights. This is sometimes termed *multimodal data visualization* [23] or *cross-media analysis* [37, p.128]. Examples for such analyses can be found in various fields, such as biomedicine [31], cancer research [24], and manufacturing [1]. These examples have in common that they each assume a rather small, hand-curated set of no more than a dozen data sources that are appropriate for the visual analysis to be conducted. Usually, this set forms a mere subset out of the large number of all available data sources, which we term *data landscape* in the spirit of similar terms for multi-dataset scenarios, such as *information landscape* [27] or *data meadow* [15]. We consider such a data landscape to be *heterogeneous*, if the individual data sources are of diverse type and structure, so that the landscape connects, for example, textual, numerical, and pictorial data sources.

If the focus of an analysis shifts, it may necessitate the incorporation of different or additional data sources from the data landscape than the ones that were originally selected. Such a flexible (re-)selection of data sources is an integral, if not the most important step of a visual analysis, as a poor choice of data to base an analysis on can render the entire analysis pointless. In contrast to this observation stands the fact that this step is hardly ever considered and most visual analysis approaches assume the data as given. Most approaches or systems support the analyst at this early stage with a mere dialog for loading the data. The few, more advanced methods are either based on statistics or authoring: the former use similarity metrics to find data sources related to given data [8], whereas the latter require a domain expert to preselect suitable data sources to pursue a given analysis goal [35]. Meta data of the data sources and dependencies between them are in both cases rarely communicated or used.

This paper presents a visualization method for giving a meaningful graphical overview of large data landscapes with potentially hundreds of different data sources to allow for an informed choice of a subset on which to pursue a subsequent visual analysis. The challenge of creating such an overview is that the necessary layout quality for serving as a useful visual index in which all the available data sources are meaningfully placed in a stable deterministic way and well separated is hard to achieve. The reason for this lies in the fact that the common graph visualization algorithms are tuned towards slightly different drawing aesthetics than those needed in this case. As a result of this, current visualizations of data landscapes are either manually designed or do not scale well beyond a few dozen datasets, as the brief overview of the related work in Section 2 shows. To overcome this challenge, we propose a carefully constructed layout concept that employs a step-wise refinement using approaches from network visualization and graph layout in Section 3. We exemplify our layout concept in Section 4 with a data landscape that was derived from the *Linking Open Data* (LOD) repository of the *Compre-*



(a) Linking Open Data Cloud [10] **(b)** Linked Open Data Graph [36]

Figure 1. Overview visualizations of the CKAN-LOD landscape. (a) depicts a manually created diagram with different domains being color-coded. According to the authors, it was incrementally crafted using the diagramming tool OmniGraffle. (b) shows an automatically laid out diagram in which the colors signify user ratings of the data sources. It was created using the Fruchterman-Reingold layout algorithm [18].

hensive Knowledge Archive Network (CKAN) containing 216 data sources and 613 interconnections between them. Finally, Section 5 concludes this paper by outlining ideas for future work.

2 Related Work

In principal, visualization across multiple data sources can occur on three levels of granularity: the *item level* of individual data records, the *table level* of different classes of data items, and the *landscape level* of entire databases. For the first two levels, with which this paper is not concerned, the reader is pointed to [9] and [11], respectively, for examples of their visualization. They both have in common that they make use of a given structure underlying and connecting the heterogeneous data for traversing it, querying it, and visualizing it. On the item level, this structure is in many cases given as an RDF graph. Whereas on the table level, the database schema is utilized as a natural structure that underlies the ensemble of tables.

On the landscape level, only a few visualizations have yet taken on the challenge of representing multiple datasets. This may be due to the fact that the advent of large data hubs and the growing Linked Open Data movement [39, ch.11] calling for visualizations on this level are rather recent developments. Such data hubs collect (links to) data sources and make meta data about them available. Notable examples are the *CKAN-LOD repository* (`http://datahub.io`), the *Socrata Social Data Platform* (`http://opendata.socrata.com`), and the *U.S. Open Government Initiative Data.gov* (`http://www.data.gov`). Together, they collect hundreds of data sources from a variety of domains and thus provide a common interface and a standardized entry point to them. Similar to the other two levels, a data landscape is usually assumed to exhibit an underlying structure – in this case an interlinkage between individual data sources as it is induced by foreign key relations between contained items. These links can be weighted by the number of individual foreign key references they subsume, and they can be orientated by the direction of these references. This structure is commonly used to employ graph-based drawing techniques for the landscape's visualization. In practice, two different such techniques are used: *manual layouts* in which the user positions all nodes interactively to produce a final static visualization and *automatic layouts* in which the positions are computed to yield a dynamically updating representation without the need for user input.

Manual layouts, such as the one given by [34] for the bioinformatics domain or the *CKAN-LOD Cloud* [10] shown in Figure 1a, tend to be very readable with a thoughtful node positioning that places data sources from the same domain close to each other. As static overviews of a data landscape, they are much less an interactive exploratory tool than a visual index of the data sources in which a click on a node redirects a web browser to the corresponding data source for further inspection. This index-like notion is underlined by the efforts to produce a meaningful relative node placement that aids in quickly (re-)finding data sources as one would desire it from an index. Furthermore, they are carefully arranged in a way so that no two nodes overlap each other, thus enabling a user to select them unambiguously. Their drawback is their inability to automatically adapt either to a change in the underlying data landscape, or to a change in the user's interest that may require a grouping by a different criteria than the domain. In both cases, the designer of the visualization has to redo the visualization and manually make the desired changes.

Automatic layouts, such as the one utilized by [24] in their *Data-View-Integrator* or the *LOD Graph* [36] shown in Figure 1b, do not suffer from this drawback. They gain their flexibility through the use of a force-based graph layout, but by using these layouts, they trade in the meaningful and well-separated placement that an index-like visualization requires. While this still works for small data landscapes with only a handful of data sources, as one can see in [24], the visual

clutter increases for larger data landscapes. At some point, the amount of overplotting makes it impossible to single out individual data sources and click on them – in particular, if they are positioned in the center of the "hairball" that the force-directed layout creates. In addition, due to their non-deterministic nature, simple force-based algorithms cannot be used to produce stable layouts, nor can they convey any semantic relation between the data sources, such as by placing them close to each other. While solutions or "work-arounds" exist to alleviate these problems – i.e., by using pinning weights to increase stability [17] and additional forces to model semantic relations, such as belonging to different domains/clusters [14] – these are so far not used in the existing data landscape visualizations.

With their respective pros and cons, these two strategies of visualizing data landscapes denote two endpoints of the same design spectrum: one being tedious to manually construct, but extremely well-suited to serve as a map of the data landscape and as a gateway to it, while the other one is easily constructed automatically, but ill-suited to perform the desired look-up task if the data landscape gets realistically large. It is thus the aim of this paper to find a compromise between the two that does no longer require manual layout work, but nevertheless serves well as the visual index of data sources that is needed. The method to achieve this is detailed in the next section.

3 A Visualization Method for Data Landscapes

The reason for the inadequacy of the existing automated approaches is the fact that the used force-directed layout algorithms are designed to optimize different aesthetic criteria than those needed for a stable visual index. For example, a meaningful placement is very hard to achieve when the node positions are used as the degree of freedom which the layout alters in order to achieve other aesthetic criteria. Among these criteria are total edge length minimization and crossing-number reduction [4], which are certainly desirable, but not at the cost of rendering the entire layout unsuitable for the task at hand. This is in line with recent results, which have shown that a layout that compromises between various aesthetics produces much better overall drawings that those that aim to achieve a few aesthetics to the fullest at the expense of all others [21]. Hence, the following section will shortly state and prioritize the layout constraints for index-like overview visualizations of data landscapes, before the concrete layout method is derived from them.

3.1 Defining the Layout Constraints

The most essential constraints imposed by the requirements of a visual index are in order of decreasing importance:

I. Stability: An essential aspect of the layout is that the same input will always result in the same output and that small changes in the input will only result in small changes in the output. Otherwise, the analyst would be presented with a different placement in each visualization session, even though the shown data landscape changed not at all or only a little.

II. No Node-Edge Overlap: As the main purpose of the layout will be to select individual nodes, it is important that the edges do not occlude them. While the edges are a fundamental part of the network, they provide only complementary information for the selection of data sources and cannot be selected themselves. Hence, edge-edge-overlap (edge crossings) can be tolerated, yet node-edge-overlap cannot.

III. No Node-Node Overlap: A similar argument can be made for the case of overlapping nodes. Since they must be selectable on an individual basis, their separation is essential to the layout.

IV. Meaningfulness: In contrast to a standard network drawing that aims at a meaningful relative positioning of nodes, the sought overview should produce a meaningful absolute positioning to aid the look-up of nodes. This implies that a change in position between visualization sessions bears meaning as well, for example, indicating that a data source's properties changed, such as its user ratings or the date of its last update.

The order of these constraints is logical as, for example, a well-separated meaningful node placement is useless if it is completely cluttered with overdrawn edges, which make it impossible to select any node at all. This implies that a less important constraint may be violated and not be achieved fully in order to fulfill a more important constraint. It can be observed that the manually generated data landscape visualizations adhere to them. For example, the LOD cloud shown in Figure 1a fulfills them in the following ways:

I. Stability is achieved through an incremental layout, which takes the last LOD cloud diagram as a starting point for including newly added data sources [10]. This way, repositioning is limited and the same data source can be found in similar positions across different versions of the diagram.

II. No Node-Edge Overlap is simply achieved by drawing the nodes on top of the edges. While it fulfills the criterion, it also hampers the attribution of edges to their incident nodes.

III. No Node-Node Overlap is achieved as a result of the manual positioning process, which aims "to form a beautiful and fluffy cloud" [10] that carefully avoids such overlap.

IV. Meaningfulness lies in this example not so much in the absolute position of a node, but rather in the relative position with respect to its neighboring nodes belonging to the same application domain. This is additionally highlighted by the color-coding of the different domains (cp. Figure 1a).

This shows that manually generated landscape visualizations come already pretty close to what we want to achieve with our proposed layout in an automated way. Therefore, these constraints directly influence our layout approach, as it is outlined in the next section.

3.2 Deriving our 3-Step Layout Approach

On the one hand, to the best of our knowledge, there exists no monolithic layout algorithm that fulfills the above layout constraints. On the other hand, we refrain from building a custom layout algorithm, as it would require additional implementation work to be used and it would reinvent the wheel in many aspects for which already decent solutions exist – e.g., for node overlap removal. As a compromise between these two strategies, we propose a modular step-wise layout concept that utilizes existing approaches and implementations for each step. This step-wise solution can be seen as a meta-algorithm, which does not specify in detail, which concrete method to use, but what kind of methods to use in which order. Our meta-algorithm employs one layout step per layout constraint, adapting the placement of nodes and edges so that the constraints are met.

Looking at the four layout constraints, it can be noted that **Constraint I** does not concern the layout result as much as it concerns the layout process as a whole. It is not something that can simply be imposed as an additional refinement step on an already existing layout as, for example, the third constraint can be achieved by performing a node overlap removal step. Hence, we consider the constraint of stability as a global one that must be fulfilled by all individual layout steps in order to be valid for the entire layout process.

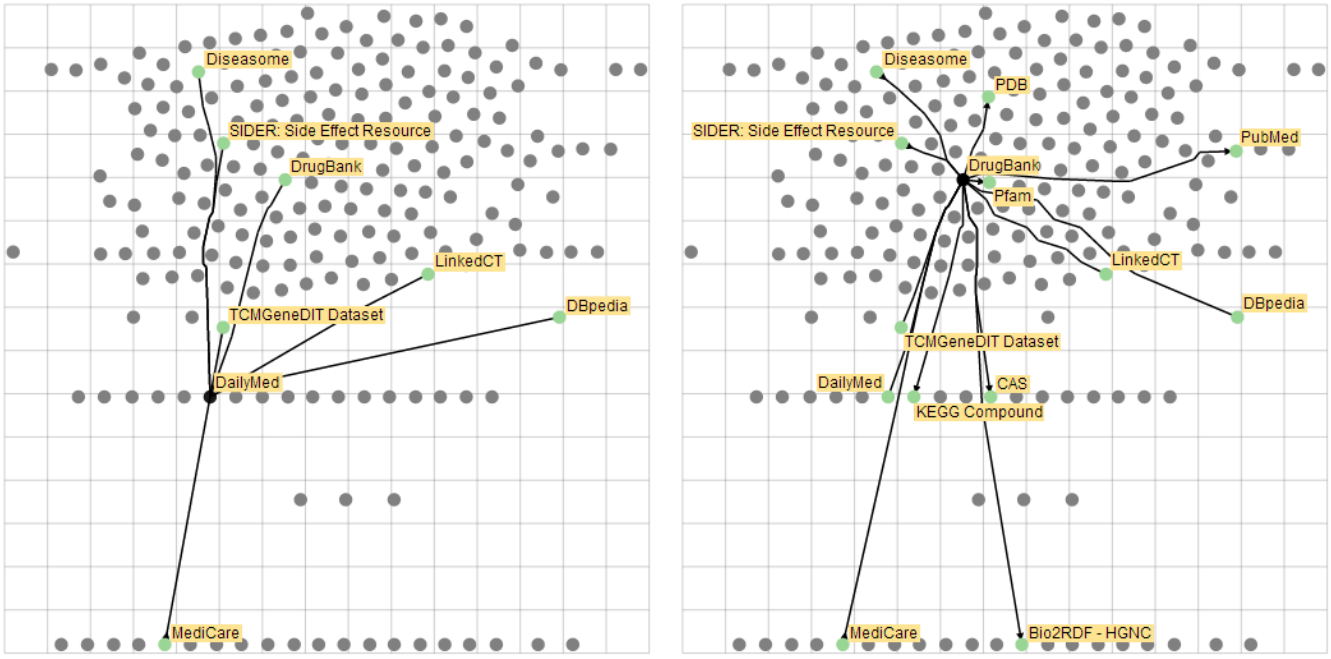
This leaves three actual layout steps to be carried out – one for ensuring each of the three remaining constraints. Since later layout steps potentially overwrite or adjust the outcome of earlier layout steps, we pursue the constraints from the least important to the most important one. This will, for example, sacrifice or reduce the meaningfulness of the node placement if this helps to remove node overlap in a later step. The three steps of our meta-algorithm are given in the following together with some concrete algorithmic suggestions for carrying them out.

Meaningful Node Placement (Constraint IV). This is the initial step that performs the placement of the nodes according to two selected numerical meta data of the data sources in a Cartesian coordinate system, similar to the GraphDice technique [5] or to the Semantic Substrates [33]. For example, the nodes can be placed on the X-axis according to the size of a data source and on the Y-axis according to the time of their last update. This way, all large and recently updated data sources will be placed at the top-right of the layout, whereas smaller and older data sources are placed towards its bottom-left. It is obvious that such a mapping of the nodes onto the X/Y-plane is deterministic. Depending on what is important for the analysis at hand, different meta data can be used. For instance, the time of the last update may be useful in financial or clinical scenarios where it is important to always work with the most recent data available. Whereas the number of incoming edges (number of other data sources referring to a data source) could be used together with the average user rating as an indicator of the trustworthiness of a data source, if a scenario depends on such a notion.

Node Overlap Removal (Constraint III). As the initial step places nodes with the same meta data on the same X/Y-position, the resulting overlap has to be removed in order to permit their individual selection. Unfortunately, many common node overlap removal approaches are either not deterministic (e.g., force-directed adjustments, such as [25]) or they interfere too strongly with the meaningful placement (e.g., [26]). Ideally, overlapping nodes are distributed in the vacant vicinity of where the overlap occurs. Suitable approaches are, for example, based on stress majorization, such as the *Fast Node Overlap Removal* [12, 13]. As a result of this step, all nodes have their distinct drawing space and a minimum distance to each other, while still being in the proximity of their original position and thus remain quick to locate. With this step, the nodes are separated and the edges can be routed through the gaps the node overlap removal created between them.

Edge Routing (Constraint II). In contrast to many established network layout methods, the node placement is in our case not driven by connectivity, i.e., nodes that are connected are not necessarily placed close to each other. Hence, it is important to show the edges in order to make the connections between the data sources explicit. This raises the problem of edges overplotting the nodes, which could be circumvented by following the idea of the LOD Cloud and drawing them underneath the nodes. Yet, this makes it hard to attribute edges to a particular pair of nodes if they cross underneath a number of them. This effect is not as visible in Figure 1a, because the data sources are grouped by their domain. Since intradomain cross-references are much likelier than interdomain relations, edges in this diagram are in general rather short and thus less prone to cross a lot of nodes. Since this does not hold true for every node placement strategy, we are required to route around the nodes. This problem can traditionally be reduced to finding shortest paths in the visibility graph of the layout. A reasonably fast algorithm building on this reduction is given in [28], yielding an almost Flow Map-like edge routing [29]. Furthermore, it is also possible to show edges only on demand for nodes of interest, if the density of the data landscape calls for it.

As these three steps of the meta-algorithm merely define the intention and the constraints of the operation to be performed, they do not require a particular fixed technique to carry them out. Instead, depending on the application case, different techniques than the ones suggested in the above description can be plugged into these steps. For example, if more than two meta data shall be mapped to the X/Y-plane, Multidimensional Scaling (MDS) [38, 7] can be used. Whereas, if interactivity is needed, a more scalable heuristic for node overlap removal, such as PRISM [16], can be applied. The following section will exemplify the utilization of our layout method for the CKAN-LOD.



(a) DailyMed and related data sources **(b)** DrugBank and related data sources

Figure 2. Overview visualization of the CKAN-LOD landscape with 216 data sources and 613 links, laid out by their *user rating* (y-axis) and *size* (x-axis). In both examples, a pharmaceutical database is selected (black) and their related data sources are highlighted (green).

4 Applying our Method to the CKAN-LOD Landscape

This paper uses the CKAN-LOD repository as an example data landscape, as it is not only freely available, but also because it is the one which is probably the best publicized and researched data landscape. The structure of this landscape has been previously described in [6] and [20, ch.3], and thoughtfully been analyzed in [30]. It is particularly well suited to exemplify our visualization approach, as it is up to date the only data landscape for which manual and automated layout solutions exist (cp. Figure 1).

We implemented our layout approach using the algorithms mentioned in the previous section to perform the individual steps. It is realized as a Java software that retrieves and parses the CKAN-LOD repository and generates an SVG diagram with added JavaScript interactivity. This way, the layout can be computed offline (e.g., in regular intervals as a cron job on the server) and its result is then published online at a fixed URL upon its completion. We found this to be a more useful setup than a client-side layout, as parsing the repository and generating the layout requires a few minutes, which a user may not be willing to wait. The layout time of a few minutes also reflects the compromise we made between the existing fast but cluttered automatic force-based layouts and the time-consuming but tidy manual layout generation.

The result can be seen in Figure 2 that depicts the subset of the CKAN-LOD landscape, for which the used numerical meta data *average rating* and *number of items* were available. To yield an uncluttered view of the data landscape, we display links only on demand for selected data sources (black). Their directionality is signified by an arrowhead that is placed at the far end of each edge by the related data source (green). If the arrowhead points away from the selected data source towards the related one, it indicates that the selected data source references the related one. If it points towards the selected data source, the selected data source is referenced by the related one. If no arrowhead is shown, the dependency is bidirectional.

In the case at hand, an analyst seeks a pharmaceutical data source, which is known to be a challenging endeavor in the field of health care and life sciences [32]. Currently, mainly list-based approaches are used to get an overview of various pharmaceutical data sources [22]. The analyst starts with an obvious candidate, the well-known DailyMed database shown in Figure 2a. Its neighborhood of dependent data sources looks promising, as it links out to databases with further information about diseases (Diseasome), clinical trials (LinkedCT), Traditional Chinese Medicine (TCMGeneDIT), and side effects (SIDER), which will ease investigations in any of these particular aspects. A double-click on the data source opens up the DailyMed webpage where the analyst discovers a serious drawback to this particular data source: it provides its data mainly in the form of English text that is hard to parse automatically (see [3]) and to distill into visualizations. Hence, she seeks for an alternative and finds it in the neighborhood of the DailyMed database: the DrugBank shown in Fig. 2b. Not only does the DrugBank provide its information in a more structured manner, but it also links to all the same databases plus some important additional ones, such as PubMed for publications, the Bio2RDF Pfam database of protein families, and the KEGG compound database. Its apparent higher appeal is furthermore reflected in its higher average user rating, which puts it more towards the top of the landscape.

To confirm that the higher average user rating does not reflect mere subjective experiences, the analyst can

(a) DailyMed and related data sources

(b) DrugBank and related data sources

Figure 3. Visualizations of the same data sources as in Figure 2, but now laid out by their *user rating* (y-axis) and *incoming connections* (x-axis).

switch to a different layout of the data landscape, which positions the data sources by user rating and the number of incoming connections. The number of incoming connections reflects how many other data sources rely on a given data source and thus acts as a proxy for its general dependability and credibility. The result of this switch is shown in Figure 3 and the analyst can see from it at a first glance that both data sources – DailyMed and DrugBank – are placed at the same horizontal position and are thus equal with respect to incoming connections. A closer look reveals that actually both data sources are linked to by the very same set of seven related data sources.

As a result, the analyst was able to identify a more suitable data source from the overview visualization and to confirm by changing the layout that a switch from DailyMed to DrugBank will not make any subsequent pharmaceutical analysis less reliable.

5 Conclusion and Future Work

The proposed overview visualization for data landscapes provides a valuable method for selecting data sources for their subsequent visual analysis. It achieves this by not simply running a standard algorithm, but by fulfilling a thoughtfully prioritized list of aesthetic constraints that are targeted directly towards the task of looking up data sources. As a result, it yields a visual index of data sources that could only be produced manually before.

In future work, we aim to explore two directions. The first is based on the observation that our visualization method is only as powerful as the available numerical meta data for the meaningful positioning of the data sources in the visual index. The more of these meta data can be provided, the more ways of mapping out the data landscape are possible and thus the better it can be tailored to the needs of the analyst. Additional meta data can either be retrieved from dedicated providers, such as the LODStats project [2], or be derived using services, such as GeoIP lookups to fetch the lat/lon-coordinates of the database locations. The latter would for example allow us to map the data sources onto a world map and thus tie the abstract geography of the data landscape to the physical geography of the globe.

The second direction to explore is to provide a more seamless navigation experience when switching between the overview visualization of the landscape that serves as a substitute for the File-Open-dialog and the actual visualization of a selected data source. For example, it is conceivable to allow for zooming into regions of the landscape visualization and as the zoomed-in nodes have more screen space available, they are substituted by portals in which they are visualized in more detail, similar to the approach proposed by [19]. This way, the user would not have to leave the visualization to switch to a different dataset, but simply zoom-out to the topmost overview of the entire data landscape and then zoom back into a different dataset. This would effectively tie these two levels of visualization closer together – something that would be unthinkable with a mere File-Open-dialog.

Acknowledgements

The authors gratefully acknowledge funding from the German Research Foundation (DFG).

REFERENCES

[1] M. Aehnelt, H.-J. Schulz, and B. Urban. Towards a contextualized visual analysis of heterogeneous manufacturing data. In *ISVC'13: Proceedings of the Interna-*

tional Symposium on Visual Computing, pages 76–85. Springer, 2013.

[2] S. Auer, J. Demter, M. Martin, and J. Lehmann. LODStats – an extensible framework for high-performance dataset analytics. In A. ten Teije, J. Völker, S. Handschuh, H. Stuckenschmidt, M. d'Acquin, A. Nikolov, and N. A.-G. N. Hernandez, editors, *EKAW'12: Proceedings of the International Conference on Knowledge Engineering and Knowledge Management*, Lecture Notes in Computer Science, pages 353–362. Springer, 2012.

[3] C. Barrière and M. Gagnon. Drugs and disorders: From specialized resources to web data. In *Proceedings of the Workshop on Web Scale Knowledge Extraction*, 2011.

[4] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[5] A. Bezerianos, F. Chevalier, P. Dragicevic, NiklasElmqvist, and J.-D. Fekete. Graphdice: A system for exploring multivariate social networks. *Computer Graphics Forum*, 29(3):863–872, June 2010.

[6] C. Bizer. The emerging web of linked data. *IEEE Intelligent Systems*, 24(5):87–92, September 2009.

[7] A. Buja, D. F. Swayne, M. L. Littman, N. Dean, H. Hofmann, and L. Chen. Data visualization with Multidimensional Scaling. *Journal of Computational and Graphical Statistics*, 17(2):444–472, June 2008.

[8] J. Caldas, N. Gehlenborg, A. Faisal, A. Brazma, and S. Kaski. Probabilistic retrieval and visualization of biologically relevant microarray experiments. *Bioinformatics*, 25(12):i145–i153, May 2009.

[9] M. Cammarano, X. Dong, B. Chan, J. Klingner, J. Talbot, A. Halevy, and P. Hanrahan. Visualization of heterogeneous data. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1200–1207, 2007.

[10] R. Cyganiak and A. Jentzsch. Linking Open Data cloud diagram. `http://lod-cloud.net/`, September 2011.

[11] D. M. de Lima, J. F. Rodrigues Jr., and A. J. M. Traina. Graph-based relational data visualization. In *IV'13: Proceedings of the International Conference Information Visualisation*. IEEE Computer Society, 2013.

[12] T. Dwyer, K. Marriott, and P. J. Stuckey. Fast node overlap removal. In P. Healy and N. S. Nikolov, editors, *GD'05: Proceedings of the International Symposium on Graph Drawing*, Lecture Notes in Computer Science, pages 153–164. Springer, 2005.

[13] T. Dwyer, K. Marriott, and P. J. Stuckey. Fast node overlap removal – correction. In M. Kaufmann and D. Wagner, editors, *GD'06: Proceedings of the International Symposium on Graph Drawing*, Lecture Notes in Computer Science, pages 446–447. Springer, 2006.

[14] P. Eades and M. L. Huang. Navigating clustered graphs using force-directed methods. *Journal of Graph Algorithms and Applications*, 4(3):157–181, 2000.

[15] N. Elmqvist, J. Stasko, and P. Tsigas. DataMeadow: a visual canvas for analysis of large-scale multivariate data. *Information Visualization*, 7(1):18–33, February 2008.

[16] E. Emden and Y. Hu. Efficient, proximity-preserving node overlap removal. *Journal of Graph Algorithms and Applications*, 14(1):53–74, 2010.

[17] Y. Frishman and A. Tal. Multi-level graph layout on the GPU. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1310–1319, November-December 2007.

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

[19] S. Hadlak, H.-J. Schulz, and H. Schumann. In situ exploration of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2334–2343, 2011.

[20] T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan&Claypool Publishers, 2011.

[21] W. Huang, P. Eades, S.-H. Hong, and C.-C. Lin. Improving multiple aesthetics produces better graph drawings. *Journal of Visual Languages and Computing*, 24(4):262–272, 2013.

[22] A. Jentzsch, O. Hassanzadeh, C. Bizer, B. Andersson, and S. Stephens. Enabling tailored therapeutics with linked data. In C. Bizer, T. Heath, T. Berners-Lee, and K. Idehen, editors, *LDOW'09: Proceedings of the WWW'09 Workshop on Linked Data on the Web*, volume 538 of *CEUR Workshop Proceedings*, 2009.

[23] J. Kehrer and H. Hauser. Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, 2013.

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

[25] W. Li, P. Eades, and N. Nikolov. Using spring algorithms to remove node overlapping. In S.-H. Hong, editor, *APVIS'05: Proceedings of the Asia Pacific Symposium on Information Visualization*, Conference in Research and Practice in Information Technology, pages 131–140. Australian Computer Society, 2005.

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

[27] V. L. O'Day and R. Jeffries. Orienteering in an information landscape: how information seekers get from here to there. In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel, and T. N. White, editors, *INTERCHI'93: Proceedings of the INTERACT'93 and CHI'93 Conference on Human Factors in Computing Systems*, pages 438–445. ACM Press, 1993.

[28] M. H. Overmars and E. Welzl. New methods for computing visibility graphs. In *SCG'88: Proceedings of the Symposium on Computational Geometry*, pages 164–171. ACM Press, 1988.

[29] D. Phan, L. Xiao, R. Yeh, P. Hanrahan, and T. Winograd. Flow map layout. In J. Stasko and M. O. Ward, editors, *InfoVis'05: Proceedings of the IEEE Symposium on Information Visualization*, pages 219–224. IEEE Computer Society, 2005.

[30] M. A. Rodriguez. A graph analysis of the Linked Data Cloud. *arXiv.org e-print service*, 0903.0194v1, March 2009.

[31] H. Rohn, C. Klukas, and F. Schreiber. Creating views on integrated multidomain data. *Bioinformatics*, 27(13):1839–1845, June 2011.

[32] M. Samwald, A. Jentzsch, C. Bouton, C. S. Kallesøe, E. Willighagen, J. Hajagos, M. S. Marshall, E. Prud'hommeaux, O. Hassenzadeh, E. Pichler, and S. Stephens. Linked open drug data for pharmaceutical research and development. *Journal of Cheminformatics*, 3(1):19, 2011.

[33] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006.

[34] S. Stephens, D. LaVigna, M. DiLascio, and J. Luciano. Aggregation of bioinformatics data using Semantic Web technology. *Web Semantics: Science, Services and Agents on the World Wide Web*, 4(3):216–221, September 2006.

[35] M. Streit, H.-J. Schulz, A. Lex, D. Schmalstieg, and H. Schumann. Model-driven design for the visual analysis of heterogeneous data. *IEEE Transactions on Visualization and Computer Graphics*, 18(6):998–1010, 2012.

[36] E. Summers and R. Cyganiak. Linked Open Data graph. `http://inkdroid.org/lod-graph/`, November 2010.

[37] J. J. Thomas and K. A. Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society, 2005.

[38] W. S. Torgerson. Multidimensional scaling: I. Theory and method. *Psychometrika*, 17(4):401–419, December 1952.

[39] L. Yu. *A Developer's Guide to the Semantic Web*. Springer, 2011.