Eurographics Conference on Visualization (EuroVis) (2012)
M. Meyer and T. Weinkauf (Editors)

Sankey Arcs - Visualizing edge weights in path graphs

Till Nagel[1,2,3], Erik Duval[2], Andrew Vande Moere[2], Kristian Kloeckl[1] and Carlo Ratti[1]

[1]MIT Senseable City Lab, USA
[2]KU Leuven, Belgium
[3]Interaction Design Lab, FH Potsdam, Germany

Abstract

Arc diagrams allow exploring relations and their strength between sequential nodes. Previous solutions suffer from displaying all arcs at the center of a node, which can lead to visual obstruction. We present a new technique, which extends the arc diagram technique by laying out the weighted edges of a node adjacent to each other. The aim of our Sankey Arc technique is to improve clarity, to enable users perceiving and comparing weighted edges in path graphs. The technique is illustrated using a dataset on travel paths in a public transit network.

Categories and Subject Descriptors (according to ACM CCS): D.2.2 [Software]: Design Tools and Techniques—Flow charts I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Arc diagrams are an established method to visualize relations between nodes in a simple path graph, which are laid out in one dimension. They have the ability to display multivariate node data, as well as edge properties. Weighted arc diagrams display the strength of relations, revealing the relative and proportional connectivity weight between connected nodes, as well as clusters of connections between neighborhoods.

Weighted arc diagrams represent connection weight as arc thickness, and display all arcs of the same node on top of each other. This may lead to a seemingly complex graphical structure, thus concealing some of the information encoded in the diagram. Furthermore, overlaid arcs of varying width tend to overemphasize nodes with a few strong connections.

One solution is to encode in- and out-weights in an additional glyph, for instance as bar graphs or – more compact – as node size. While this facilitates comparing the nodes and their properties, it does not allow comparing the edges, and introduces further clutter.

In this paper we present Sankey Arcs, a novel visualization technique to visualize relation quantities between ordered nodes. The contribution of this work is an extension to the existing technique, in an aim to display edge weights with less obstruction. It builds upon arc diagrams, which visualize weighted connections in simple linear graphs, by dis-

DOI: 10.2312/PE/EuroVisShort/EuroVisShort2012/055-059

playing vertex strength as well as the total weights of incoming (in-weight) and outgoing edges (out-weight). This enables users to perceive and compare weighted edges in path graphs.

Vertex strength measures the weight of nodes in terms of the total weight of their relations [BBPSV04]. Analogous, in-weight measures the total weight of all incoming, and out-weight the total weight of all outgoing connections.

In this paper, we identify the design goals, describe the implementation, and discuss the advantages and limitations of our technique. As a proof of concept, we present how we employed Sankey Arcs in a visualization of public transit data.

2. Related Work

Wattenberg introduced arc diagrams as a visualization technique for highlighting repeated subsequences in sequential data such as text or music [Wat02]. Since then, arc diagrams have been applied in different domains, from e-mail threads [Ker03], to command chains [Aut10], to influence relations [DCW11].

The technique also has been extended in various ways. It has been expanded to cover other data structures (e.g. to visualize hierarchical data [NSC05], [GBD09]), visually modified to encode additional properties (e.g. color-coded to show categories [Fry06]), and enhanced by additional functionality (e.g. interactive filtering [CISSW06]). In their survey on visualization techniques, Heer et al. describe arc diagrams more generally as "one-dimensional layout of nodes, with circular arcs to represent links" [HBO10].

Arc diagrams have been used to visualize weighted graphs, in which the strength of a connections is encoded as arc thickness (e.g. to represent common terms in text documents [CISSW06], or hotel guests [WFR*07]). In the PivotGraph technique, thickness represents the number of aggregated edges [Wat06], which, in the special case of a one-dimensional layout, is equivalent to an arc diagram. The last visualization only displays one arc per node, and thus has no overlappings, while the first two display multiple connections as transparent arcs on top of each other. Some Circos variant displays weighted links between circle segments as ribbons with varying thickness, with the ribbons spread out on the circle segments [KSB*09]. Conceptually, our approach is relatively similar, yet our approach is linear instead of circular, and seems more suitable for sequential datasets.

3. Sankey Arcs

Our Sankey Arc technique displays all weighted edges of a node adjacent to each other. The development was based on the need to visually adjust nodes with similar vertex strength, and to overcome occlusion of arcs at the node base. The intention of this technique is to reduce visual clutter in comparison with classic arc diagrams.

The design of Sankey Arcs was guided by the following goals:

1. Display node strength independent of edge distribution.
2. Enable comparison of in-weight and out-weight.
3. Emphasize the total weight of a node.

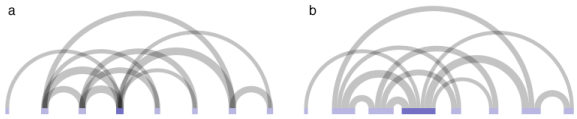

Figure 1: *Two nodes with same vertex strength in a) a classic arc, and in b) a Sankey Arc diagram.*

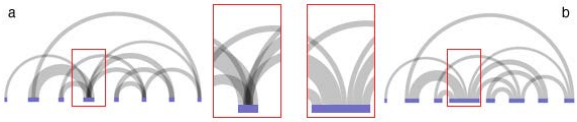

Figure 2: *Transparency vs spread out.*

By spreading out the imposts (i.e. arc bases), all edge weights of a node become visible. In Figure 1b a node with many thin arcs is visually comparable to another node having few strong connections. In a diagram with overlapping arcs Figure 1a this is concealed.

The volume of stacked transparent arcs also obscure the in-weights and out-weights (Figure 2a). Bundling the incoming and outgoing arcs helps understanding at a glance which direction has more weight (Figure 2b). Furthermore, by showing each edge weight in parallel, the total weight is integrated into the arcs. Oppose to some disjoint glyph, this displays the value in a unified manner (with the cost of using more space).

Besides enabling to visually compare nodes and edges, arc diagrams support understanding the relation between weight and distance (as the angle of the arc hints to the distance), and the distribution of amounts (e.g. if there are clusters of connected nodes). These two features already exist in classic arc diagrams, but are more profoundly visually highlighted in Sankey Arcs.

3.1. Algorithm

Figure 3: *Same graph as a) unordered arc diagram, with b) spread out arcs, and c) reordered head and tail positions.*

In order to minimize arc crossings, we propose the application of two additional rules: Firstly, all incoming arcs (i.e.

arcs where the current node is the tail node) are placed on the side of their head nodes, and all outgoing arcs on the side of their tail nodes (Figure 3b). Secondly, the edges of a single node are ordered by distance towards their respective counter nodes (Figure 3c). That is, an incoming arc is placed the further towards the head node the closer that head node is (respectively, outgoing arcs towards tail nodes).

As the arc's start position is depending on all arcs at the head node, (and analogous its end position on all at the tail node), the algorithm has to iterate twice over all edges in order to incorporate the position and weight of every incoming and outgoing edge.

The general algorithm to create Sankey Arcs is described as pseudo-code. Nodes are required to be sorted by their natural order (time, spatial distance, etc), or by some artificial order (e.g. alphabetical).

```
For each edge e
  e.head.outWeight += e.weight
  e.tail.inWeight += e.weight
  For both e.nodes n
    n.totalWeight += e.weight
Sort all edges by head and then by tail nodes
For each edge e
  // If e is edge of new head node
  If e.head != prevEdge.head
    headPos = e.head.pos + e.head.totalW/2
  headPos -= e.weight
  // If e is first edge of tail node
  If tailPos in posList for e.tail
    tailPos = e.tail.pos + e.tail.totalW/2 -
      e.tail.outW
  tailPos.x -= e.weight
  drawArc(headPos, tailPos, e.weight)
```

For simplicity the pseudo-code describes the algorithm for arcs from left to right. We also used the edge weights directly, while in implementations they should be encoded as thickness via some appropriate mapping method

3.2. Positioning nodes

In comparison to a classic weighted arc diagram (Figure 4a), the nodes in a Sankey Arc diagram tend to consume more space. This is due to the width nodes with more than one arc need to display the total weight of their connections. Thus, nodes might converge or overlap, and arcs of neighboring nodes might obstruct each other (Figure 4b). To resolve this, the design properties of the diagram need to be modified. We describe two possible methods.

- **Adapting the layout** algorithm to position the nodes. This can be achieved by either enlarging the distance between each node (e.g. using more than half the maximum width of the largest node) (Figure 4c), or by using a fixed gap between nodes. In both cases, the overall size of the diagram increases.

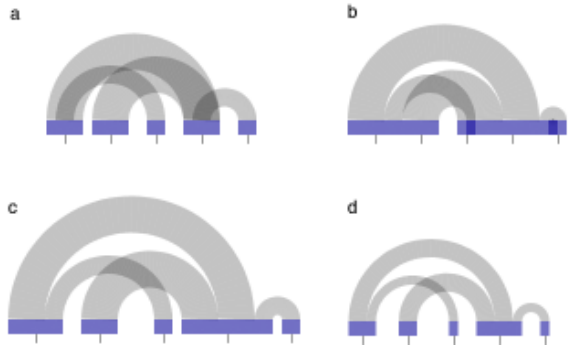

Figure 4: *Graph shown as a) arc diagram, and as Sankey Arcs with b) overlapping nodes, c) larger distance between nodes, and d) reduced arc thickness.*

- **Modifying the visual encoding** for arc thickness. Applying a mapping method producing thinner arcs and nodes results in a total width equal to or smaller than the original (Figure 4d). In the example shown, the adaptation was done in a way so that the same distance between nodes (as in Figure 4a) could be used.

Both are feasible solutions to prevent overlapping. The choice of which one to use depends on the size of the data set and the number and weights of the connections. Generally, we recommend to use a combination of both, starting with reducing the distance between the nodes, and – if that is not sufficient – adapting the visual encoding iteratively until the whole diagram fits into the designated space.

4. Limitation

In the classic technique all arcs start at the center of a node so that the height of an arc only depends on the distance between its nodes. Hence, a viewer comparing two arcs can directly deduce the distance equivalence or difference from the height (Figure 5a). In Sankey Arc diagrams the starting and ending positions of an arc rely on all other arcs of the same node. Thus, it not always encodes an equal distance to an equal height (Figure 5b).

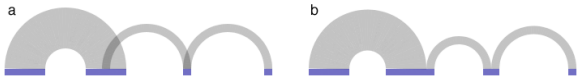

Figure 5: *Comparison of a) an arc diagram with same heights, and b) a Sankey Arc with different heights.*

Nodes of an arc diagram are positioned equidistantly in most cases. Often, arc diagrams are visualizing data sets with discrete order (such as website ranking [Fry06], or text chapters [HBO10]). Dörk et al. explicitly opted against a continuous timeline for their arc diagram even though the nodes are

representing years, because of the non-uniform distribution of their data [DCW11]. In theses cases, an exact comparison of height might not be necessary. This limitation affects arc diagrams where the precise distance is important, and is really relevant only for nodes with very high vertex strength.

5. Case study

As a proof of concept we applied the technique in a visualization for public transit ridership in Singapore (Figure 6). This was part of the LIVE Singapore project, which combines and disseminates various urban data sets to provide citizens visual and tangible access to information about their city [KSDLR11].

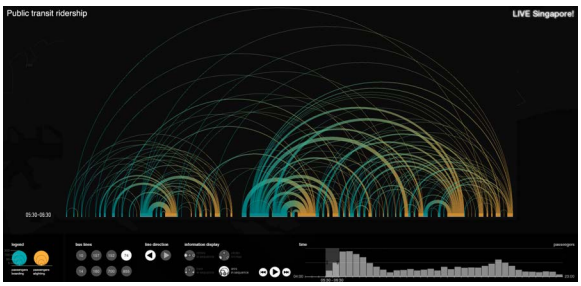

Figure 6: *Ridership visualization with legend, bus lines and visualization controls, and an interactive time range slider.*

In Singapore, a contact-less smart card is used to pay the distance-based fare. There is a strong incentive for passengers to not only tap in on boarding, but also to tap out when alighting, as otherwise the maximum fare has to be paid. This results in precise origin-destination paths for each ride.

The arc diagrams visualize boarding and exiting passengers at bus stops, and the rides taken in between. Each arc signifies passengers going from one station to another. The width of an arc represents the number of passengers. Users can switch between multiple bus lines, and interactively select a time range (see Figure 6), to compare for instance the morning with the evening rush hour.

In Figure 7, the bus stops of (the Western part of) bus line 10 in Singapore are shown. The number of passengers are double encoded as arc transparency, so that the most taken bus rides are easily discernible. Now, bus stops which are public transit hubs become visible. Some hubs have few strong connections (e.g. 15), some have many weak ones (e.g. 32), some have a mix of both (e.g. 23).

The visualization properties used in Figure 7a overemphasize strong connections. Figure 7b shows bar graphs for each bus stop, with the upper one representing boarding, and the lower one alighting passengers. As the bars show, bus stops 1 and 32 have similar quantities of boarding passengers. This becomes apparent only with the Sankey Arcs technique (Figure 7c).

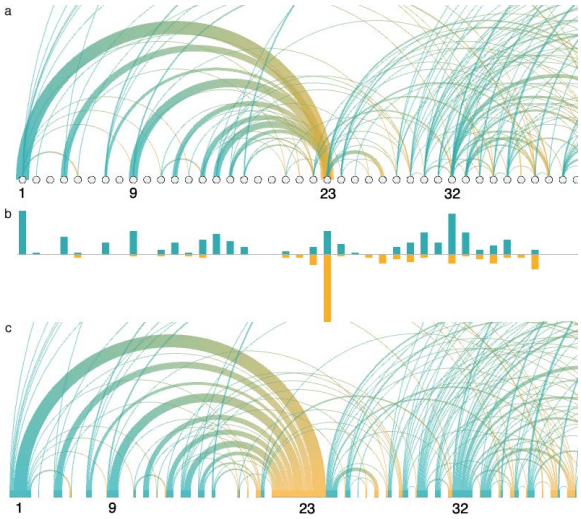

Figure 7: *Rides between bus stops as classic arc diagram (top) and as Sankey Arc (bottom), with passengers as bar diagram (middle).*

The ride direction is represented as color gradient from turquoise (boarding) to orange (alighting). While the results of a user study by Holten and van Wijk [HvW09] suggest to use tapered connections for direction, we could not apply a visual mapping with varying thickness as this was used for encoding the weight (cf [HIvWF11]). Hence, we used the second best according to the original study, and chose a color schema with higher legibility for color-blind viewers [JK07].

6. Conclusions

The main contribution of this paper is a visualization technique to display edge weights of nodes in a path graph. We described how this extension of arc diagrams create less visual obstruction by spreading out arcs.

With the use case we have demonstrated that Sankey Arcs help comparing nodes and their edge weights. Applying the technique in a real world visualization helped us to validate the applicability, from displaying arc directionality with colors, applying it to a set of public transit data with vastly different properties, and testing the performance with a highly responsive implementation for an interactive time range slider.

Our hypothesis as set forth in this paper and that will need experimental verification is that Sankey Arcs outperform classic arc diagrams especially in cases where few strong connections represented by arc width visually overlap with many weaker connections leading to an overall decrease in readability of the diagram. We plan a user study to establish in quantitative terms which technique of arc diagrams performs better.

References

[Aut10] AUTODESK RESEARCH: Command usage arc diagrams, 2010. URL: http://www.autodeskresearch.com/pages/infovis/command_usage_arc. 2

[BBPSV04] BARRAT A., BARTHELEMY M., PASTOR-SATORRAS R., VESPIGNANI A.: The architecture of complex weighted networks. *PNAS 101*, 11 (March 2004), 3747–3752. 2

[CISSW06] CHEN C., IBEKWE-SANJUAN F., SANJUAN E., WEAVER C.: Visual analysis of conflicting opinions. In *IEEE VAST* (2006), Wong P. C., Keim D. A., (Eds.), IEEE, pp. 59–66. 2

[DCW11] DÖRK M., CARPENDALE S., WILLIAMSON C.: Edgemaps: Visualizing explicit and implicit relations. In *Proceedings of Visualization and Data Analysis 2011 (VDA 11)* (2011), vol. 7868. 2, 4

[Fry06] FRY B.: Linkology, Feb. 2006. URL: http://benfry.com/linking/. 2, 3

[GBD09] GREILICH M., BURCH M., DIEHL S.: Visualizing the Evolution of Compound Digraphs with TimeArcTrees. *Computer Graphics Forum 28*, 3 (2009), 975–982. 2

[HBO10] HEER J., BOSTOCK M., OGIEVETSKY V.: A tour through the visualization zoo. *Commun. ACM 53* (June 2010), 59–67. 2, 3

[HIvWF11] HOLTEN D., ISENBERG P., VAN WIJK J. J., FEKETE J.-D.: An extended evaluation of the readability of tapered, animated, and textured directed-edge representations in node-link graphs. In *PacificVis* (2011), Battista G. D., Fekete J.-D., Qu H., (Eds.), IEEE, pp. 195–202. 4

[HvW09] HOLTEN D., VAN WIJK J. J.: A user study on visualizing directed edges in graphs. In *CHI* (2009), Jr. D. R. O., Arthur R. B., Hinckley K., Morris M. R., Hudson S. E., Greenberg S., (Eds.), ACM, pp. 2299–2308. 4

[JK07] JENNY B., KELSO N. V.: Color design for the color vision impaired. *Cartographic Perspectives 58* (2007), 61–67. 4

[Ker03] KERR B.: Thread arcs: An email thread visualization. In *Proceedings of the IEEE Symposium on Information Visualization* (2003), IEEE Computer Society, pp. 211–218. 2

[KSB*09] KRZYWINSKI M. I., SCHEIN J. E., BIROL I., CONNORS J., GASCOYNE R., HORSMAN D., JONES S. J., MARRA M. A.: Circos: An information aesthetic for comparative genomics. *Genome Research* (2009). 2

[KSDLR11] KLOECKL K., SENN O., DI LORENZO G., RATTI C.: Live singapore!- an urban platform for real-time data to program the city. In *Computers in Urban Planning and Urban Management, CUPUM 2011* (Alberta, Canada, July 2011). 4

[NSC05] NEUMANN P., SCHLECHTWEG D. S., CARPENDALE S.: ArcTrees: Visualizing Relations in Hierarchical Data. Brodlie K., Duke D., Joy K., (Eds.), Eurographics Association, pp. 53–60. 2

[Wat02] WATTENBERG M.: Arc diagrams: Visualizing structure in strings. In *Proceedings of the IEEE Symposium on Information Visualization* (2002), Wong P. C., Andrews K., (Eds.), IEEE Computer Society, pp. 110–116. 2

[Wat06] WATTENBERG M.: Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI conference on Human Factors in computing systems* (2006), ACM Press, pp. 811–819. 2

[WFR*07] WEAVER C., FYFE D., ROBINSON A., HOLDSWORTH D., PEUQUET D., MACEACHREN A. M.: Visual exploration and analysis of historic hotel visits. *Information Visualization 6*, 1 (2007), 89–103. 2