Drawing Algorithms for Linear Diagrams

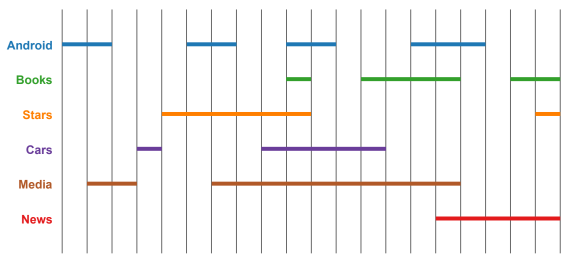
Peter Chapman, Kevin Sim and Huanghao Chen

Edinburgh Napier University, UK
p.chapman@napier.ac.uk,
k.sim@napier.ac.uk,
40506146@live.napier.ac.uk

Abstract. It is known that drawing linear diagrams with few line segments produces cognitively effective diagrams. However, methods to minimise the number of line segments in a diagram have not been investigated. In this Abstract, we report on four different approaches to drawing linear diagrams: creating a drawing order based on sets; creating a drawing order based on set-intersections; simulated annealing; and a transformation to another problem, in this case the Travelling Salesman Problem (TSP). We tested each of these approaches on a corpus of 440 linear diagrams with a range of characteristics. The algorithm which produced the lowest number of line segments in a diagram was based on the Lin-Kernighan algorithm for solving instances of TSP. The second contribution of the work is the corpus of 440 linear diagrams, which can be used as a benchmark set of problems for others who seek other solutions to the problem of drawing linear diagrams.

Keywords: linear diagrams, minimisation algorithms, benchmark results

1 Introduction



Linear diagrams have been shown to be an effective representation of set-based data in a number of different comparisons [1, 3]. The figure to the left shows a linear diagram. Each vertical space (hereafter overlap) represents a non-empty intersection in the underlying sets. For example, the tenth overlap in the figure above contains lines for all sets except News. This overlap encodes the information that the intersection Android∩Books∩Stars∩Cars∩ Media∩(not News) is non-empty. Note that the horizontal ordering of the overlaps does not alter the underlying information. Sets can be drawn with more than one *line segment*. For example, Android is drawn with four line segments, wheras News is drawn using only one. A collection of drawing guidelines were developed in [5], and of chief interest for this Abstract is the guideline that one should "*draw linear diagrams with a minimal number of line segments*". In [5], a drawing algorithm was given, but with no guarantees as to its performance. In this work, we focus on producing and evaluating algorithms for drawing linear diagrams, where the primary metric used for evaluation is the number of line segments produced. The input for each algorithm is a collection of overlaps, and the output is an ordering of those overlaps. A goal of this work is to improve existing tools for drawing diagrams, and thus a secondary consideration is the ease with which an algorithm can be implemented.

2 Candidate Algorithms

Four different approaches to drawing linear diagrams were investigated. The first two started from first principles: a **set-based** approach, and an **overlap-based approach**. In addition, we also investigated using **simulated annealing**, and a **transformation to a known problem**.

With the set-based approach, a drawing order for sets was determined based on how many set intersections it contained. For example, if set A had non-empty intersections with 10 other sets, and set B only had non-empty intersections with 5 other sets, then the drawing algorithm would prioritise drawing set A over set B. A consequence of this ordering is that it is more likely that set A would be drawn in fewer segments than set B. **Three** variant algorithms based on this approach were developed.

With the overlap-based approach, a drawing order for overlaps was determined based on similarity between overlaps. The *distance* between two overlaps is defined as the number of sets they have in common [5]. The algorithms then select the closest remaining overlap to be appended (or pre-pended, depending on the particular algorithm) to an existing diagram. **Four** variants of this approach were used, of which one is the algorithm found in [5].

The distance between two overlaps also informs the transformation to the *Travelling Salesman Problem* (TSP). Each overlap is treated as a city, and the cost of travelling between a pair of cities is taken as the distance between the two relevant overlaps. We thus transform our drawing problem into an instance of TSP on a complete graph. An application of the Lin-Kernighan algorithm [2] follows, whose solution we transform back into a linear diagram. *Simulated annealing* has been used as a strategy for solving many different combinatorial optimisation problems [4]. The ordering of overlaps in a linear diagram is such a problem.

3 Methodology and Results

A collection of 440 linear diagrams was produced, which varied in size from 5 sets and 10 overlaps, to 50 sets and 70 overlaps. Each algorithm was applied to each test diagram[1]. A clear ranking was determined across the test set:

$$\text{TSP} \succ \text{Simulated annealing} \succ \text{Overlap approach}^{\star} \succ \text{Set approach}^{\star}$$

For the overlap and set approaches, only the best for each approach was tested against TSP and Simulated annealing. The particular overlap-based algorithm which performed best was not that which was implemented in [5], but rather a simpler variant of it. That the TSP-solution is best is not surprising: a large body of research exists on providing efficient and effective solutions to the TSP. The TSP approach provides an approximate 6% improvement on the set-intersection approach, in terms of the number of line segments in the resulting diagram. For the largest diagrams, this corresponds to diagrams that use around 50 fewer line segments. In terms of effect sizes, Cohen's d ranges from 2.35 (TSP vs. set approach) to 0.20 (TSP vs. simulated annealing).

4 Conclusions

By comparing a variety of drawing algorithms on a large corpus of test cases, we have found that an efficient TSP-solver can outperform algorithms which directly attempt to draw a linear diagram. However, the complexity of the TSP-solver restricts its applicability, whereas the best set-intersection-based algorithm can (and has) been implemented in Javascript within a web-browser. This ease of access can lower the barriers to using linear diagrams as a visualisation technique.

In addition, we have provided 440 test diagrams, and a benchmark set of results. Any further algorithms can be tested on these diagrams, and thus be easily compared with the results of this paper. All diagrams, datasets and analysis is available at `https://doi.org/10.17869/enu.2021.2748170`.

References

1. Chapman, P., Stapleton, G., Rodgers, P., Michallef, L., Blake, A.: Visualizing sets: An empirical comparison of diagram types. In: Diagrams 2014, pp. 146–160. Springer (2014)
2. Helsgaun, K.: General k-opt submoves for the Lin–Kernighan TSP heuristic. Mathematical Programming Computation 1(2), 119–163 (2009)
3. Luz, S., Masoodian, M.: A comparison of linear and mosaic diagrams for set visualization. Information Visualization 18(3), 297–310 (2019)
4. Nahar, S., Sahni, S., Shragowitz, E.: Simulated annealing and combinatorial optimization. In: 23rd ACM/IEEE Design Automation Conference. pp. 293–299 (1986)
5. Rodgers, P., Stapleton, G., Chapman, P.: Visualizing sets with linear diagrams. ACM Transactions on Computer-Human Interaction (TOCHI) 22(6), 27 (2015)

[1] Simulated annealing was run 30 times on each diagram, and the mean number of line segments was determined.