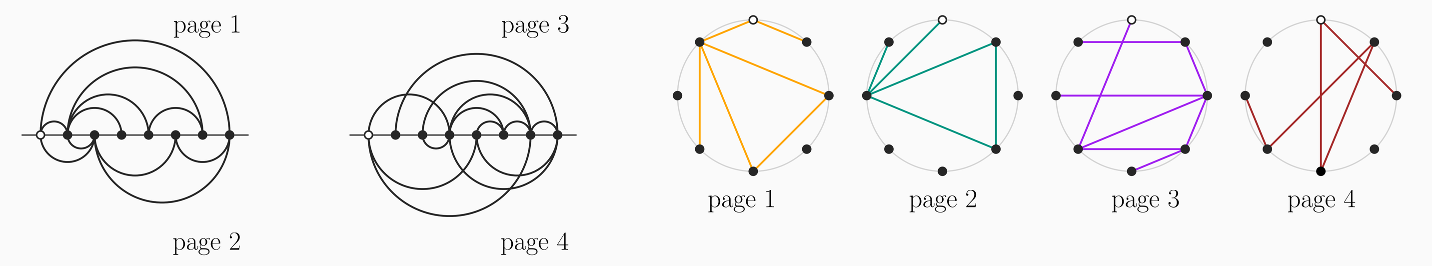
Heuristic Picker for Book Drawings

Jonathan Klawitter and **Tamara Mchedlidze**
Karlsruhe Institute of Technology • Germany

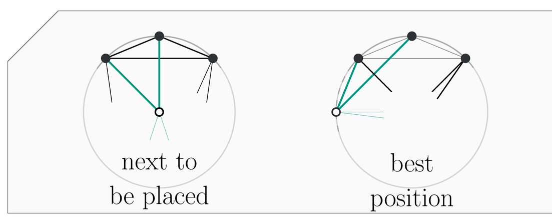
Abstract The problem of crossing minimization in *k-page book drawings* is in general $\mathcal{NP}$-hard [1]. Thus, it is common to use heuristics. Among those, simple heuristics presented in literature, compute a vertex order and an edge distribution for a book drawing independently.
In this poster, we present several new simple heuristics, including *full drawing heuristics*, that compute vertex order and edge distribution at the same time, and their experimental evaluation together with the most prosiminig simple heuristcs from the literature.
Besides the heuristcs considered in this poster, there exist other simple heuristics that were outperformed by the presented ones. There also exist more complex heuristics, based on neural networks [7,11,13,16], simulated annealing and evolutionary techniques [2,5,8,14,15], which are out of the focus of this poster.



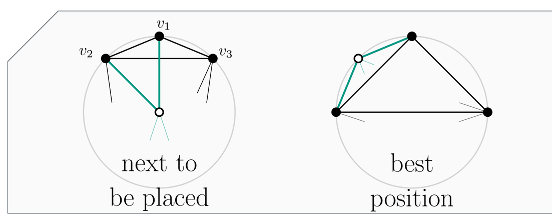
Four page book drawing illustrated with *linear* spine.

Four page book drawing illustrated with *circular* spine.

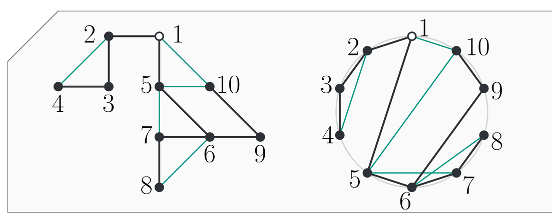
Vertex order heuristics (operate in one page)



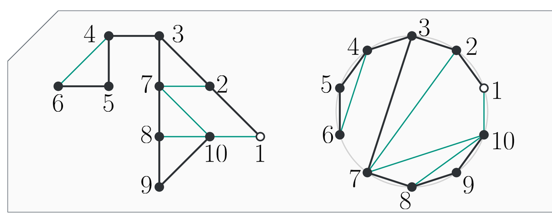
`conCro`: Builds the vertex order step by step. At each step it selects the vertex with the most already placed neighbours (connectivy → `con`) and places it on one of the two ends of the current spine where it introduces fewer crossings with unplaced edges (→ `Cro`) [3].



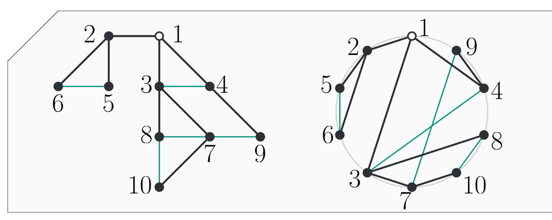
`conGreedy`: Like `conCro` it builds the vertex order step by step. At each step it selects the vertex with the most already placed neighbours, however, it places the vertex on the position of the current spine where it introduces fewer crossings with already placed edges.



`randDFS`: The vertices are ordered based on a DFS traversal of the graph that starts at a random vertex and randomly selects the next vertext to visit [2].



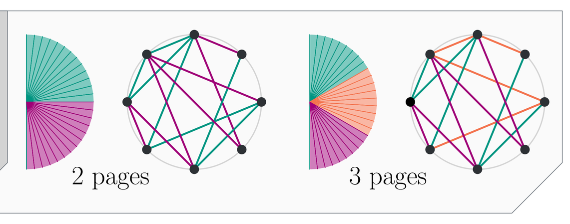
`smlDgrDFS`: Similarly to `randDFS`, the vertices are ordered based on DFS traversal, but it starts at the vertex of the smallest degree and selects the next vertices by increasing degree [6].



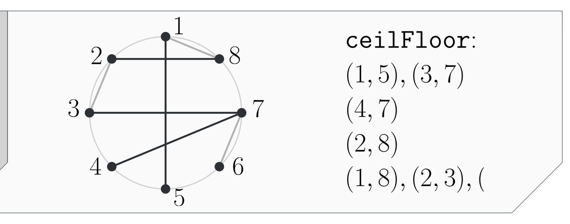
`treeBFS`: Orders vertices based on a crossing-free 1-page book drawing of a BFS spanning tree.

Edge distribution heuristics

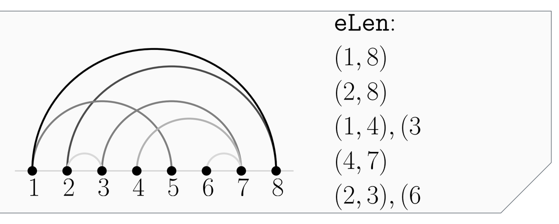
`slope`: Distributes edges based on their slope in a circular drawing [10]. Perfoms well on dense and perfect on complete graphs in some cases [12].



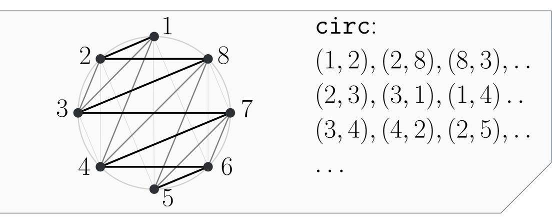
`ceilFloor`: Distributes edges greedily in the order of descending length in a circular drawing [15].



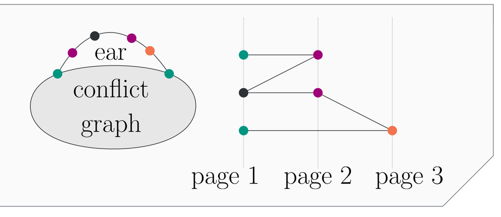
`eLen`: Like `ceilFloor`, it distributes the edges greedily in the order of descending distance of their endpoints on the linear spine [4]. The original motivation was to place first the edges with highest probability to create crossings achieved instead by `ceilFloor`.



`circ`: Distributes edges greedily in the order that produces book embeddings of complete graphs on their pagenumber [15]. The order is a set of paths, zig-zaging on a circular spine.



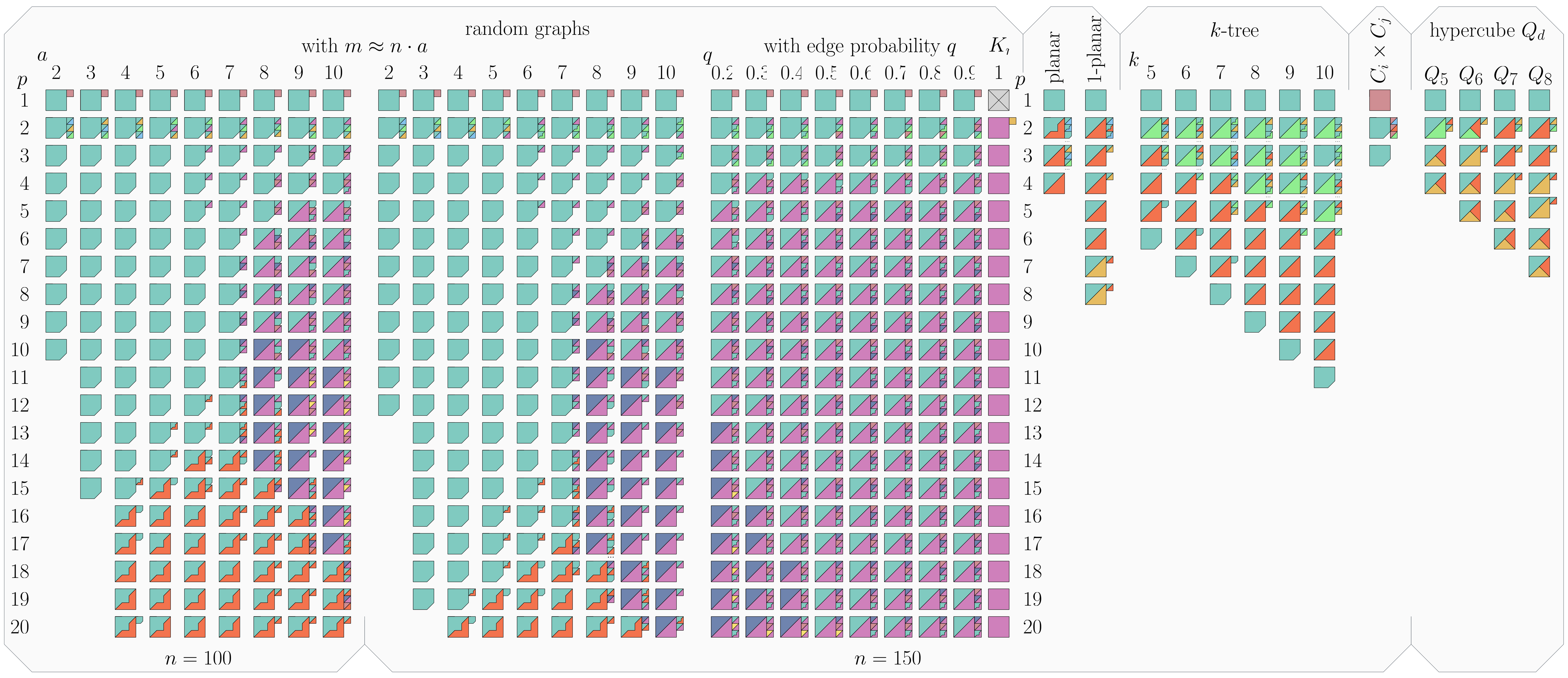
`earDecomp`: Constructs the conflict graph of the edges in a circular drawing, and an ear decomposition of this conflict graph, and then alternate the vertices of each ear (edges of the original graph) between two or three pages.



Full drawing heuristics

`conGreedy+`: This full drawing heuristic works like `conGreedy`, but distributes an edge to the best page greedily as soon as it gets *closed*. In contrast to `smlDgrDFS+` and `randDFS+`, this directly affects the computation of the rest of the vertex order. Thus it can also be used as improved vertex order heuristic by discarding the edge distribution afterwards.

Full drawing heuristics compute vertex order and edge distribution at the same time. For example, following the idea by He et al. [9] `smlDgrDFS`, `randDFS` and `conGreedy` can be extended to distribute an edge to the best page greedily as soon as it gets *closed*, i.e. at the moment its second end-vertex appears on the spine. The corresponding full heuristics are called `smlDgrDFS+`, `randDFS+` and `conGreedy+`.



The figure on the left illustrates the evaluation of the heuristics on different graph classes and different number of pages p. Big tiles represent the heuristic combination or full drawing heuristic that performed best in terms of average number of crossings, computed for 200 instances for the spefic case. The upper-left part of a tile is coloured acording to vertex order heuristic, and the bottom-right according to edge distribution heuristic; it is subdivided into two parts if two heurstics perform equally.

The small tiles on the right of a big tile represent heuristics following in the performance, from top to bottom, if their performance is at most 10% below the winning one.

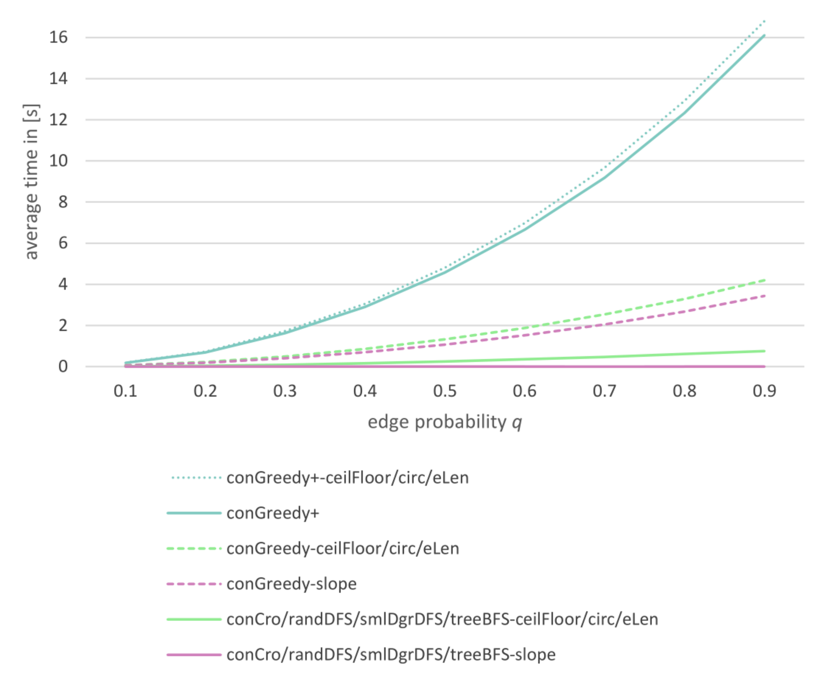
Our test suite contained among others two sets of random graphs: with a linear number of edges, $m \approx na$, and with a quadratic number of edges (given by edge probability q in the Erdős-Reni model), as well as planar, 1-planar, k-tree, cycle product and hypercubes. See [12] for the complete results of the experiment.

Discussion

It can be seen from the image above that the best heuristic combination depends not only on the density of the graphs, but remarkably also on the structural properties of the graphs. For example, the combination `conGreedy+-ceilFloor` performs best on planar and 1-planar graphs, while `conGreedy+`, as full drawing heuristic, performs best on random graphs with the same density.

We observe that our first extension of the vertex order heuristc `conCro` to `conGreedy` , produces results with fewer crossings. The second extension of `conGreedy` to the full drawing heuristic `conGreedy+` sometimes achieves even fewer crossings. However, both these extensions come with the cost of higher running time, which was clearly noticeable in the experiments (e.g., see figure on the right).

Furthermore, we could observe that `conGreedy+-ceilFloor/eLen` achieved crossing-free book drawings of hypercubes Q_d when $p =$ pagenumber (tested up to $d = 10$).



The figure on the left shows the average running time as a function of the density, for random graphs with quadratic number of edges, $n = 150$ and four pages.

References.

1. M. J. Bannister and D. Eppstein. Crossing minimization for 1-page and 2-page drawings of graphs with bounded treewidth. *CoRR*, 2014.
2. R. Bansal, K. Srivastava, K. Varshney, N. Sharma, et al. An evolutionary algorithm for the 2-page crossing number problem. In *Evolutionary Computation*, IEEE, 2008.
3. M. Baur and U. Brandes. Crossing reduction in circular layouts. In *Graph-Theoretic Concepts in Computer Science*, Springer, 2005.
4. R. Cimikowski. Algorithms for the fixed linear crossing number problem. *Discrete Applied Mathematics*, 122(1), 2002.
5. H. He, M. Newton, and O. Skora. Genetic algorithms for bipartite and outerplanar graph drawings are best! Communications of the 31st Conference on Current Trends in Theory and Practice of Informatics, 2005.
6. H. He and O. Skora. New circular drawing algorithms. *Conference Papers and Presentations (Computer Science)*, 2004.
7. H. He, O. Skora, and E. Makinen. An improved neural network model for the two-page crossing number problem, *IEEE Transactions on Neural Networks*, 17(6), 2006.
8. H. He, O. Skora, and E. Mkinen. Genetic algorithms for the 2-page book drawing problem of graphs. *Journal of heuristics*, 13(1), 2007.
9. H. He, O. Skora, A. Salagean, and I. Vrto. Heuristic crossing minimisation algorithms for the two-page drawing problem. 2006.
10. H. He, O. Skora, and I. Vrto. Crossing minimisation heuristics for 2-page drawings. *Electronic Notes in Discrete Mathematics*, 22, 2005.
11. H. He, A. Slgean, E. Mkinen, and I. Vrto. Various heuristic algorithms to minimise the two-page crossing numbers of graphs. *Open Computer Science*, 5(1), 2015.
12. J. Klawitter. Algorithms for crossing minimisation in book drawings. Masters thesis, Karlsruhe Institute of Technology, 2016.
13. D. Lpez-Rodrguez, E. Mrida-Casermeiro, J. Ortz-de Lazcano-Lobato, and G. Galn- Marn. K-pages graph drawing with multivalued neural networks. In *Artificial Neural Networks ICANN 2007*, Springer Berlin Heidelberg, 2007.
14. T. Poranen, E. Mkinen, and H. He. A simulated annealing algorithm for the 2-page crossing number problem. In *Proceedings of International Network Optimization Conference (INOC)*, 2007.
15. D. Satsangi, K. Srivastava, and G. Srivastava. K-page crossing number minimiza- tion problem: An evaluation of heuristics and its solution using gesakp. *Memetic Computing*, 5(4), 2013.
16. J. Wang. Hopfield neural network based on estimation of distribution for two-page crossing number problem. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 55(8), 2008.