Heuristic Picker for Book Drawings
Jonathan Klawitter and Tamara Mchedlidze
Karlsruhe Institute of Technology u Germany

Abstract
The problem of crossing minimization in k-page book drawings is in general 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 paper, we present several new simple heuristics, including full drawing heuristics, that compute vertex order and edge distribution at the same time, and experimentally evaluate together with promising simple heuristics from the literature.

Besides the heuristics considered in this paper, 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 [5,9,14,15], which are out of the focus of this paper.

Vertex order heuristics (operate in one page)

- **comCrt**: Builds the vertex order step by step. At each step it selects the vertex with the most already placed neighbours (connectivity $\rightarrow$ com) and places it on one of the two ends of the current spine where it introduces fewer crossings with unplaced edges ($\rightarrow$ Cro) [6].
- **comGreedy**: Like comCrt & 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 vertex to visit [2].
- **smlDgrDFS**: Similar 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 [5].
- **tredDFS**: Orders vertices based on a crossing-free k-page book drawing of a DFS spanning tree. 

Edge distribution heuristics

- **algs**: Distributes edges based on their slope in a circular drawing [10]. Performs well on dense and perfect graphs.
- **ceilFloor**: Distributes edges greedily in the order of descending length of their endpoints on the linear spine [4]. The original motivation was to place first the edges with highest probability to create crossings achieved through after ceilFloor.
- **slen**: Like ceilFloor, it distributes the edges greedily in the order of descending distance of their endpoints on the linear spine [4].
- **circ**: Distributes edges greedily in the order that produces book embeddings of complete graphs on their page number [15]. The order is a set of paths, zig-zagging on a circular spine.

Full drawing heuristics

- **comGreedy**+ This full-drawing heuristic works like comGreedy, 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.
- **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.

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 comGreedy+ceilFloor performs best on planar and 1-planar graphs, while comGreedy+ performs best on random graphs with the same density.

We observe that our first extension of the vertex order heuristic comCrt to comGreedy, produces results with fewer crossings. The second extension of comGreedy to the full drawing heuristic comGreedy+ sometimes achieves even fewer crossings. However, both these extensions come with the cost of higher running time, which was clearly noticeable in the experiments.

Furthermore, we could observe that comGreedy+ceilFloor/slen achieved crossing-free book drawings of hypercubes $Q_k$ when $p =$ page number (tested up to $d = 10$).

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 comGreedy 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 resulting full heuristics are called smlDgrDFS, randDFS and comGreedy.

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 heuristics and full drawing: full drawing heuristics or full drawing heuristic that performed best in terms of average number of crossings, computed for 200 instances for the specific case. The upper-left part of a tile is colored according to vertex order heuristic, and the bottom-right according to edge distribution heuristic: it is subdivided into two parts if two heuristics perform equally.

The small tiles on the right of a big tile present simple heuristics from the literature.

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
3. B. Chandra, M. J. Bannister, D. Eppstein, and J. Vitek. crossings vs. edge length for complete graphs in some cases [12].