Computational Geometry 52 (2016) 18–33

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Edge routing with ordered bundles [☆]

Sergey Pupyrev [c,d], Lev Nachmanson [b], Sergey Bereg [a,*], Alexander E. Holroyd [b]

[a] *Department of Computer Science, University of Texas at Dallas, USA*
[b] *Microsoft Research, USA*
[c] *Department of Computer Science, University of Arizona, USA*
[d] *Institute of Mathematics and Computer Science, Ural Federal University, Russia*

ARTICLE INFO

Article history:
Received 19 September 2012
Accepted 22 October 2015
Available online 27 October 2015

Keywords:
Graph drawing
Edge routing
Edge bundling
Crossing minimization

ABSTRACT

Edge bundling reduces the visual clutter in a drawing of a graph by uniting the edges into bundles. We propose a method of edge bundling that draws each edge of a bundle separately as in metro-maps and call our method ordered bundles. To produce aesthetically looking edge routes, it minimizes a cost function on the edges. The cost function depends on the ink, required to draw the edges, the edge lengths, widths and separations. The cost also penalizes for too many edges passing through narrow channels by using the constrained Delaunay triangulation. The method avoids unnecessary edge–node and edge–edge crossings. To draw edges with the minimal number of crossings and separately within the same bundle, we develop an efficient algorithm solving a variant of the metro-line crossing minimization problem. In general, the method creates clear and smooth edge routes giving an overview of the global graph structure, while still drawing each edge separately and thus enabling local analysis.

1. Introduction

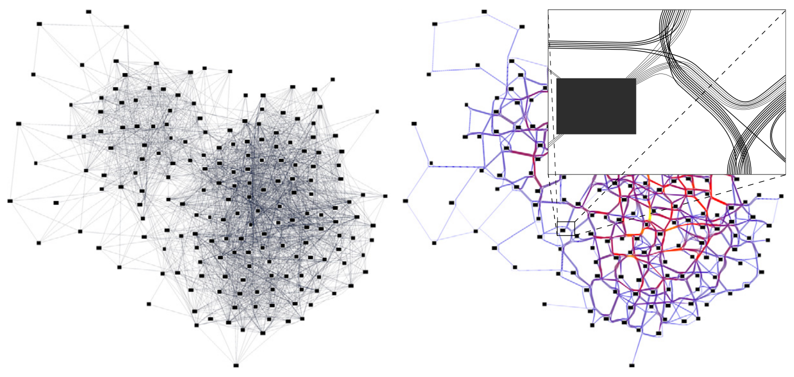
Graph drawing is often used in information visualization for exploring and analyzing network data. The core components of most graph drawing algorithms are the computation of positions of the nodes and edge routing. In this paper we concentrate on the latter problem.

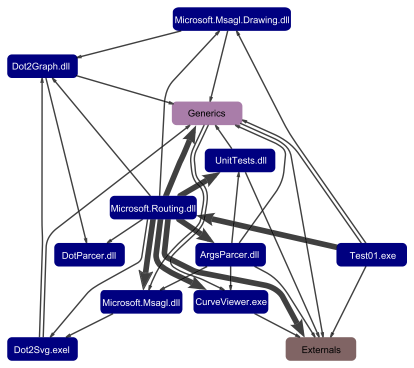
For many real-world graphs with substantial numbers of edges traditional algorithms produce visually cluttered edge routes. The relations between the nodes are difficult to analyze by looking at such drawings. Recently, edge bundling techniques have been developed in which some edge segments running close to each other are collapsed into bundles to reduce the clutter [23,28,29]. While these methods create an overview drawing, they typically allow the edges within a bundle to cross and overlap each other arbitrarily, making individual edges hard to follow. In addition, previous approaches allow the edges to overlap the nodes and obscure their text or graphics [11,19,23,29,30].

We present a novel edge routing algorithm for undirected graphs that produces drawings in a “metro map” style; see Fig. 1. The graphs for which our algorithm is best applicable are of medium size with a large number of edges, although it can process larger graphs efficiently too.

[☆] A part of the results of this paper was presented at the *19th International Symposium on Graph Drawing* [38].
[*] Corresponding author.
E-mail addresses: spupyrev@gmail.com (S. Pupyrev), levnach@microsoft.com (L. Nachmanson), besp@utdallas.edu (S. Bereg), holroyd@microsoft.com (A.E. Holroyd).

http://dx.doi.org/10.1016/j.comgeo.2015.10.005
0925-7721/© 2015 Elsevier B.V. All rights reserved.


(a)



(b)

Fig. 1. (a) Visualizations of `jazz` graph with straight-line edges (left) and ordered edge bundles (right). (c) An example of ordered edge bundles where edges walk around node labels.

The input for our algorithm is an undirected graph with given node positions. These positions can be generated by a graph layout algorithm, or in some applications (for instance, geographical ones) they are fixed in advance. During the algorithm the node positions are not changed. The main steps of our algorithm are similar to existing approaches, but with several innovations, which we indicate with *italic text* in the following description.

Path routing. The edges are routed along the edges of some underlying graph and the overlapping parts are organized into bundles. One approach here has been to minimize the total "ink" of the bundles. However, this often produces excessively long paths. *For this reason we introduce a novel cost function for edge routing.* Minimizing this function forces the paths to share routes, creating bundles, but at the same time it keeps the paths relatively short. Additionally, the cost function penalizes routing too many edges through narrow gaps between the nodes. *We furthermore show how to route the bundles around the nodes.*

Path construction. In this step the paths belonging to the same bundle are "nudged" away from each other. The effect of this action is that individual edges become visible and the bundles acquire thickness. *Contrary to previous approaches, we try to draw the edges of a bundle as parallel as possible with a given gap.* However, such a routing may not always exist because of the limited space between the nodes. We provide a heuristic that finds a drawing with bundles of a suitable thickness.

Path ordering. To route individual edges, an order of the edge segments within each bundle needs to be computed. This order minimizes the number of crossings between the edges of the same bundle. The problem of finding such an order is a variant of the metro-line crossing minimization problem. *We provide a new efficient algorithm that solves this problem exactly*.

The next section summarizes related work. In Section 3 we introduce the ordered edge bundling algorithm, followed by the detailed explanation of its steps in Sections 4–6. Results of experiments are presented in Section 7. We discuss some additional issues and future work in Section 8.

2. Related work

Edge routing. The problem of drawing edges as curves has been considered in several visualization papers. Such edges, with their smoother and organic shape, have an aesthetic appeal to humans, who are known to prefer round shapes [3]. Curves can improve performance in some typical graph reading tasks when compared to straight-line edges [20,40,42]. In some scenarios, graph nodes have associated text labels, and curved edges are used to avoid node interiors; see Fig. 1(b). Dobkin et al. [14] initiated a visibility-based approach in which the shortest obstacle-avoiding route for an edge is chosen. The method requires the computation of the visibility graph, which can be too slow for graphs with hundreds of nodes. An improvement was suggested by Dwyer and Nachmanson [17], achieving faster edge routing using sparse visibility graphs. The latter method constructs approximate shortest paths for the edges. We use the method as the starting point of our algorithm.

Dwyer et al. [16] proposed a force-directed method, which improves a given edge routing by straightening and shortening edges that do not overlap node labels. However, a feasible initial routing is needed. Moreover, the method moves the nodes, which is prohibited in our scenario.

The problem of finding obstacle-avoiding paths in the plane arises in several other contexts. In robotics, the motion-planning problem is to find a collision-free path among polygonal obstacles. This problem is usually solved with a visibility graph, a Voronoi diagram, or a combination thereof [41]. Though these methods can be used to route a single edge in a graph, they do not apply directly to edge bundling. Many path routing problems were studied in the very-large-scale integration (VLSI) community. We use some ideas of [10,12] as will be explained later.

Edge bundling. The idea of improving the quality of layouts via edge bundling is related to edge concentration [35], in which a layered graph is covered by bicliques in order to reduce the number of edges. Eppstein et al. [13,18] proposed a