EDGE OFFSET IN DRAWINGS OF LAYERED GRAPHS WITH EVENLY-SPACED NODES ON EACH LAYER

MATTHIAS F. STALLMANN

Abstract. Minimizing edge lengths is an important esthetic criterion in graph drawings. In a layered graph drawing method the total length of edges can be minimized at any of several points in the drawing process. Here we focus on edge *offset*, a measure closely related to edge length – we call it *stretch*. And we consider minimizing stretch when the permutation of nodes on each layer is determined, usually the point at which edge crossings are minimized. If we fix x-coordinates so as to distribute nodes evenly on each layer we can then permute nodes and use the permutations to assign nodes to these fixed x-coordinates with the objective of minimizing total stretch. We show that (a) the problem of minimizing stretch in this setting is NP-hard; (b) there exists a straightforward mixed integer program for stretch minimization; and (c) any heuristic or algorithm that minimizes or attempts to minimize crossings has asymptotic approximation ratio at least 2 when it comes to stretch.

1. Introduction

An ℓ-layer graph $G = (V, E)$ has $V = V_1 \oplus ... \oplus V_\ell$ and $E \subseteq \bigcup_{1 \le i < \ell}(V_i \times V_{i+1})$. In other words, the nodes are partitioned into ℓ *layers* and all edges connect nodes on adjacent layers. An *embedding* of a layered graph G defines a permutation $\pi_i : V_i \to |V_i|$. We call $\pi_i(v)$ the *position* of v. Position specifies the order of nodes on a layer but not their x-coordinates.

In the method for drawing directed graphs proposed by Sugiyama et al. and refined by others [17, 10, 11] there are five basic steps: (i) removing cycles by reversing the direction of some of the edges; (ii) assigning nodes to layers; (iii) making the layer assignment *proper* (all edges connect nodes on adjacent layers) by inserting dummy nodes; (iv) minimizing edge crossings by permuting nodes on each layer; and (possibly) (v) adjusting x-coordinates of nodes to satisfy esthetic criteria.

Prior work on reducing edge length in layered graph drawings has focused on assigning nodes to layers with minimum edge length as an objective (as suggested in Di Battista et al. [9, Ch. 9]), on permuting nodes on layers with minimum edge length as an objective (see Chimani and HungerLänder [5]), on holistic approaches that attempt to optimize multiple esthetic criteria (see, e.g., Kusnadi et al. [14] or Stolfi et al. [16]) or on adjusting x-coordinates of nodes on each layer *after* their ordering has been determined (see, e.g., Buchheim et al. [3] or Brandes and Köpf [1]).

Here we assume that layer assignment has been done and x-coordinates on each layer are given so as to distribute nodes evenly on each layer. The permutation of nodes on layers is not yet determined, as in the approach of Chimani and Hungerländer. Like them we aim to minimize is the total horizontal displacement of the edges. Their approach differs from ours in that they require nodes to occupy points with integer coordinates (without the restriction that they be evenly spaced, a restriction that requires fractional coordinates as we shall see).

Formally, if $vw \in E$, where $v \in V_i$ and $w \in V_{i+1}$, let $s(vw)$, the *stretch* of edge vw, be

$$|\tilde{\pi}_i(v) - \tilde{\pi}_{i+1}(w)|$$

where

$$\begin{array}{rcll} \tilde{\pi}_k(x) & = & \frac{\pi_k(x)}{|V_k|-1} & \text{if } |V_k| > 1 \\ & & \frac{1}{2} & \text{otherwise (place a single node in the center)} \end{array}$$

Note that $s(vw)+1$ is the Manhattan distance between v and w while $\sqrt{s(vw)^2+1}$ is the Euclidean distance. The stretch of an edge is referred to as the *edge offset* in [14]. The sum of the Manhattan distances of the edges is $\sum_{vw \in E} s(vw) + |E|$. However, the sum of the Euclidean distances, total edge length, is not directly related to the sum of the stretches; instead, if $d(vw)$ is the (Euclidean) length of vw, then $\sum_{vw \in E} d(vw)^2 = \sum_{vw \in E} s(vw)^2 + |E|$.

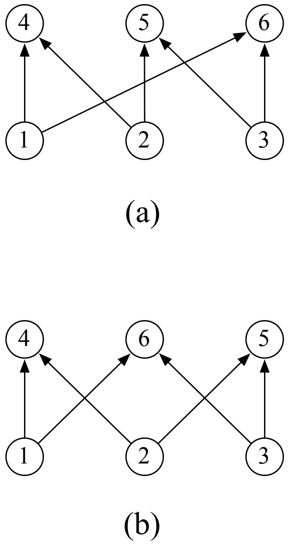


FIG. 1.1. *A two layer graph illustrating the relationship between stretch and edge length.*

For example, consider Fig 1.1. The total stretch of both embeddings is 2: In (a), edges 1-4, 2-5 and 3-6 have 0 stretch; edges 2-4 and 3-5 have stretch 0.5; and edge 1-6 has stretch 1. In (b), edges 1-4 and 3-5 have 0 stretch while edges 1-6, 2-4, 2-5 and 3-6 each have stretch 0.5. With respect to Euclidean distance, however, the embedding (a) has a total of $3 + 2\sqrt{1.25} + \sqrt{2} \approx 6.65$ while embedding (b) has a total of $2 + 4\sqrt{1.25} \approx 6.47$. The sums of the squares are as follows:

$$\begin{array}{lcll}
\sum_{vw\in E} s(vw)^2 & = & 2\cdot 0.25+1 & = 1.5 \text{ in (a)} \\
\sum_{vw\in E} d(vw)^2 & = & 3+2\cdot 1.25+2 & = 7.5 \text{ in (a)} \\
\sum_{vw\in E} s(vw)^2 & = & 4\cdot 0.25 & = 1 \text{ in (b)} \\
\sum_{vw\in E} d(vw)^2 & = & 4\cdot 1.25+2 & = 7 \text{ in (b)}
\end{array}$$

In Section 2 we show that minimizing total stretch is NP-Hard. Section 3 gives a mixed integer program for minimizing total stretch and a quadratic program for minimizing total edge length and in Section 4 we discuss the ratio between the stretch of an embedding with minimum crossings and one that has minimum stretch overall. We conclude with some remarks in Section 5.

2. NP-Completeness

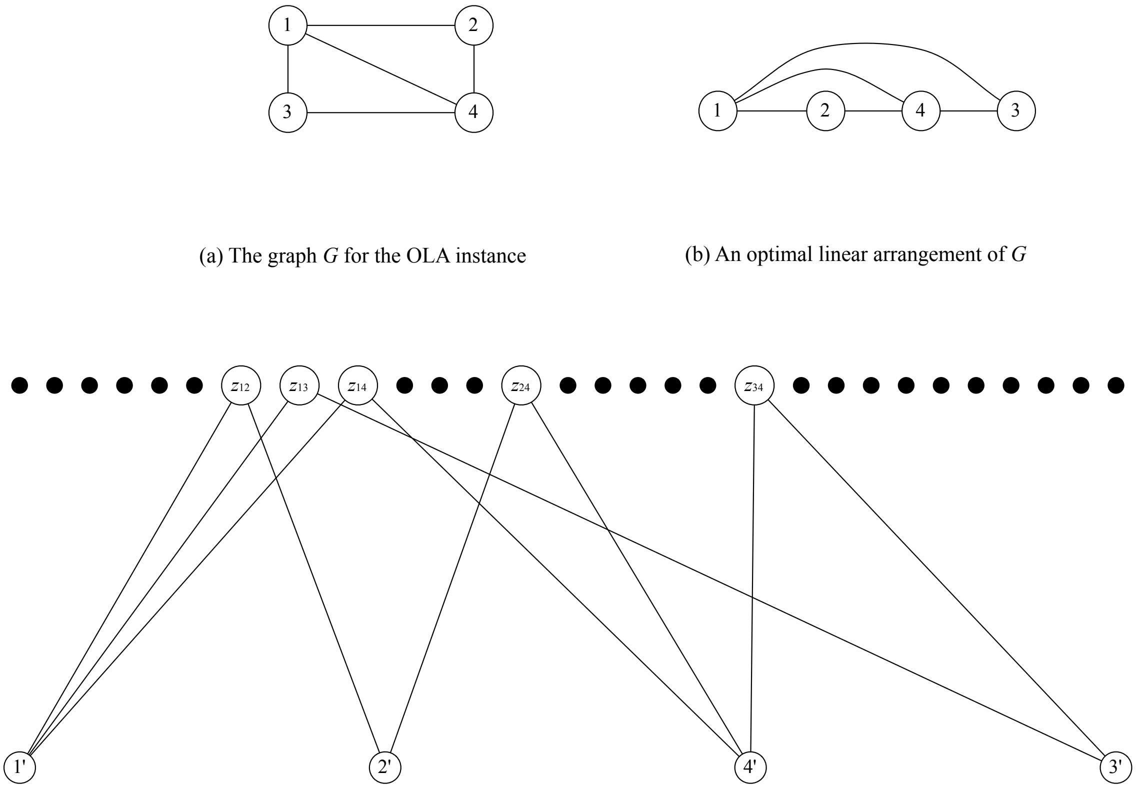
The problem of minimizing total stretch is NP-Hard, even when the graph has only two layers. Consider the following decision version of the two-layer problem.

Two-Layer Edge Stretch (TLES). Given a bipartite graph $G = (V_1, V_2, E)$ and a rational number r, do there exist bijections $\pi_1 : V_1 \to \{0, \ldots, |V_1| - 1\}$ and $\pi_2 : V_2 \to \{0, \ldots, |V_2| - 1\}$ such that

$$\sum_{vw\in E} s(vw) \le r$$

where

$$s(vw) = |\tilde{\pi}_1(v) - \tilde{\pi}_2(w)|$$



(a) The graph G for the OLA instance

(b) An optimal linear arrangement of G

(b) An embedding of G' corresponding to the linear arrangement shown in (b)

FIG. 2.1. *An example showing the reduction from OLA to TLES.*

and

$$\begin{array}{rcll} \tilde{\pi}_\ell(x) & = & \frac{\pi_\ell(x)}{|V_\ell|-1} & \text{if } |V_\ell| > 1 \\ & & \frac{1}{2} & \text{otherwise} \end{array}$$

Since total stretch can be calculated in polynomial time given a π_i for each layer i, TLES is in NP. We show that TLES is NP-Complete via reduction from Optimal Linear Arrangement (OLA) – see Garey et al. [12].

Recall the definition of OLA: Given a graph $G = (V, E)$ and a positive integer k, does there exist $f : V \to \{1, \ldots, |V|\}$ such that $\sum_{uv \in E} |f(u) - f(v)| \leq k$.

To convert an instance $\langle G = (V, E), k\rangle$ of OLA to an instance $\langle G' = (V_1, V_2, E'), r\rangle$ of TLES, let

$$\begin{array}{rcll} V_1 & = & \{v' \mid v \in V\} & \\ V_2 & = & \{z_{uv} \mid uv \in E\} & \\ & & \quad \cup\ \{y_i \mid 1 \leq i \leq n(n-1)/2 - |E|\} & \text{spacers to ensure that none of the images of the} \\ & & & \text{edges have to ``go backwards''} \\ E' & = & \{u' z_{uv} \mid uv \in E\} & \text{there are no edges involving the spacers} \end{array}$$

So, for each edge uv in the OLA instance, both u' and v' are connected to z_{uv}. The image of an edge uv of G is therefore a path $u' z_{uv} v'$. Let $r = k/(|V| - 1)$. We assume here that $|V| > 1$; otherwise the instance is trivially positive.

Suppose there exists f such that $\sum_{uv \in E} |f(u) - f(v)| \leq k$. Then we can let $\pi_1(u') = f(u) - 1$ for $u \in V$. To construct π_2 let u be the vertex of G with $f(u) = i$ and let $v_1, \ldots, v_k$ be its neighbors that have $f(v_j) > f(u)$, in arbitrary order. Then let $\pi_2(z_{uv_j}) = i(i-1)/2 + j - 1$. The spacer nodes are then assigned the remaining π_2 values.

Now consider any edge $uv \in E$ and assume $f(u) < f(v)$. First note that $\tilde{\pi_1}(u') \leq \tilde{\pi_2}(z_{uv}) < \tilde{\pi_1}(v')$. There are exactly two edges in G' that correspond to uv, namely $u'z_{uv}$ and $z_{uv}v'$. Their combined contribution to the sum is

$$\begin{aligned}(\tilde{\pi_2}(z_{uv}) - \tilde{\pi_1}(u')) + (\tilde{\pi_1}(v') - \tilde{\pi_2}(z_{uv})) &= \tilde{\pi_1}(v') - \tilde{\pi_1}(u') \\ &= (\pi_1(v') - \pi_1(u'))/(|V_1| - 1) \\ &= (f(v) - f(u))/(|V| - 1)\end{aligned}$$

Thus the sum will be $k/(|V| - 1)$.

Conversely, suppose there exist π_1 and π_2 such that $\sum_{u'z_{uv} \in E'} s(u'z_{uv}) \leq k/(|V| - 1)$. Without increasing any $s(u'z_{uv})$ we can use the spacer nodes to move the z_{uv} nodes so that $\tilde{\pi_1}(u') \leq \tilde{\pi_2}(z_{uv}) < \tilde{\pi_1}(v')$, when $\pi_1(u') < \pi_1(v')$. Then $s(u'z_{uv}) + s(v'z_{uv}) = \tilde{\pi_1}(v') - \tilde{\pi_1}(u') = (f(v) - f(u))/(|V| - 1)$. If we let $f(u) = \pi_1(u') + 1$ then $\sum_{uv \in E} |f(u) - f(v)| \leq k$.

Fig. 2.1 shows an example of the reduction. The arrangement in (b) has total cost 8. In the embedding of G' the total stretch does not depend on the exact positions of the z_{uv} as long as z_{uv} is to the right of whichever of u' or v' appears first in the embedding of V_1 and to the left of the other. In this case the edge pairs corresponding to 1-2, 1-3, 1-4, 2-4 and 3-4 have stretch values of 1/3, 1, 2/3, 1/3 and 1/3, respectively, for a total of 8/3.

3. Mixed Integer Programming Formulation

A mixed integer programming formulation for minimizing total stretch begins with the framework of an ILP for crossing minimization, such as the ones described by by Jünger and Mutzel [13], Mutzel [15], Chimani et al. [4, 7, 6] and Buchheim et al. [2], among others.

Since we are not minimizing crossings we need only those variables that keep track of positions of nodes on each layer. Let $x_{ij} = 0$ if i comes before j on a layer, 1 otherwise. and let $p_i =$ the (integer) position of i on L_i (its layer). The constraints that enforce a total order are

$$\begin{array}{rcll} x_{ij} + x_{ji} & = & 1 & \text{for } j \in L_i\text{; anti-symmetry} \\ x_{ik} - x_{ij} - x_{jk} & \geq & -1 & \text{transitivity} \\ \sum_{j \in L_i, j \neq i} x_{ji} - p_i & = & 0 & \text{where } L_i \text{ is the set of nodes in layer } i \end{array}$$

Now introduce variables s_{ij} to represent the (not necessarily integer) stretch $s(ij)$ for each edge ij. The objective is to minimize $\sum_{ij \in E} s_{ij}$. We need constraints to express the fact that

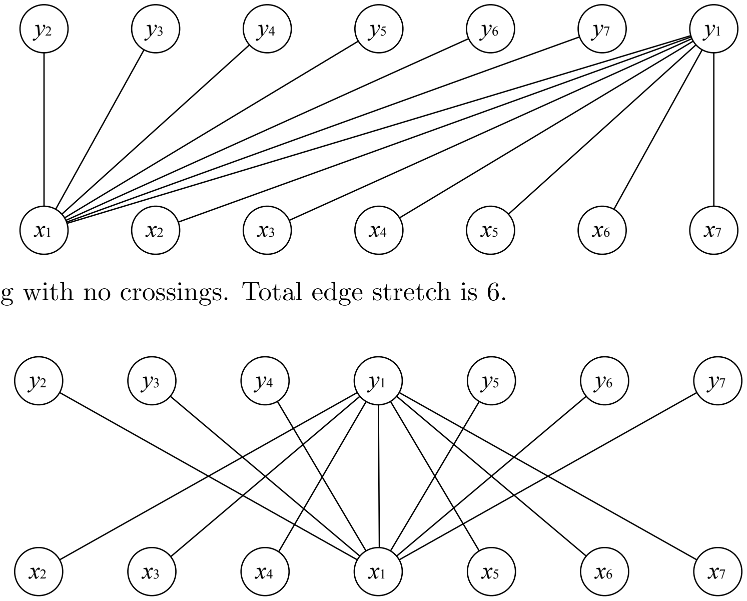
$$s_{ij} = \left| \frac{1}{|L_i| - 1} p_i - \frac{1}{|L_j| - 1} p_j \right|$$

To keep the notation simple, let

$$z_{ij} = \frac{1}{|L_i| - 1} p_i - \frac{1}{|L_j| - 1} p_j$$

(When $|L_i|$ or $|L_j| = 1$ we replace the denominator with 2 as noted earlier.) Using standard tricks for absolute value constraints, introduce binary variables b_{ij} and add the following constraints.

$$s_{ij} + z_{ij} \geq 0 \tag{3.1}$$



(a) An embedding with no crossings. Total edge stretch is 6.

(b) A embedding with minimum stretch = 4 and 18 crossings.

FIG. 4.1. *An instance ($n = 7$) of the general case of the approximation lower bound.*

$$s_{ij} - z_{ij} \geq 0 \tag{3.2}$$

$$z_{ij} + 2b_{ij} - s_{ij} \geq 0 \tag{3.3}$$

$$-z_{ij} - 2b_{ij} - s_{ij} \geq -2 \tag{3.4}$$

Constraints (3.1) and (3.2) ensure $s_{ij} \geq |z_{ij}|$. Constraints (3.3) and (3.4) ensure $s_{ij} \leq |z_{ij}|$ in the following way. We know that $-1 \leq z_{ij} \leq 1$. If z_{ij} is positive let $b_{ij} = 0$ and note that (3.3) forces $s_{ij} \leq z_{ij}$ while (3.4) fourth has no effect. If z_{ij} is negative let $b_{ij} = 1$; then (3.4) forces $s_{ij} \leq -z_{ij}$ while (3.3) has no effect.

4. Approximation Lower Bound

When it comes to approximating edge stretching, the (asymptotic) ratio between total stretch achieved by any heuristic or exact algorithm *that permutes nodes to minimize edge crossings* and minimum stretch is ≥ 2.

Consider a two layer graph $G = (V_0, V_1, E)$ with $V_0 = \{x_1, \ldots, x_n\}$ and $V_1 = \{y_1, \ldots, y_n\}$. Connect x_1 with $y_1, \ldots, y_n$ and y_1 with $x_1, \ldots, x_n$. So $E = \{x_1 y_i \mid 1 \leq i \leq n\} \cup \{x_i y_1 \mid 1 \leq i \leq n\}$.

An embedding with no crossings has x_1 on the far left and y_1 on the far right. See Fig. 4.1(a) for the example when $n = 7$. The resulting total stretch is

$$\frac{\sum_{i=1}^{n-1} i}{n-1}$$

for the edges incident on x_1 and

$$\frac{\sum_{i=1}^{n-2} i}{n-1}$$

for the remaining edges, for a grand total of $n - 1$.

The minimum total stretch is achieved by putting both x_1 and y_1 in the middle – see Fig. 4.1(b). The total stretch, if n is odd, is

$$4 \cdot \frac{\sum_{1=i}^{(n-1)/2} i}{n-1} = \frac{n+1}{2}$$

The ratio will be $2(n-1)/(n+1)$, which approaches 2.

5. Concluding Remarks

In embedding a layered graph, nodes can be permuted to minimize stretch (leading to reduced edge length) during the step whose objective is normally to minimize crossings. Some questions remain unanswered.

- The use of spacer nodes in the NP-Completeness proof make the proof less than satisfying. Is the problem really NP-Hard if we insist that the graphs are connected?
- How useful is the mixed integer program? Preliminary experiments suggest that, because of the multitude of potential total stretch values, a solver such as CPLEX [8] may take long to converge.
- Is the asymptotic lower bound of 2 on the ratio between minimum stretch when crossings are minimum and minimum stretch overall also an upper bound?

REFERENCES

[1] U. Brandes and B. Köpf, *Fast and simple horizontal coordinate assignment*, in Graph Drawing: 9th International Symposium, 2001, pp. 31–44.

[2] C. Buchheim, D. Ebner, M. Jünger, G. Klau, P. Mutzel, and R. Weiskircher, *Exact crossing minimization*, in Graphy Drawing: 14th International Symposium, vol. 3843 of Lecture Notes in Computer Science, 2006, pp. 37–48.

[3] C. Buchheim, M. Jünger, and S. Leipert, *A fast layout algorithm for k-level graphs*, in Graph Drawing, 2000, pp. 229–240.

[4] M. Chimani, C. Gutwenger, and P. Mutzel, *Experiments on exact crossing minimization using column generation*, in 5th International Workshop on Experimental Algorithms, 2006, pp. 303–315.

[5] M. Chimani and P. Hungerländer, *Multi-level verticality optimization: Concept, strategies, and drawing scheme*, Journal of Graph Algorithms and Applications, 17 (2013), pp. 329–362.

[6] M. Chimani, P. Hungerländer, M. Junger, and P. Mutzel, *An SDP approach to multi-level crossing minimization*, Journal on Experimental Algorithmics, 17 (2012).

[7] M. Chimani, P. Mutzel, and I. Bomze, *A new approach to exact crossing minimization*, in 16th Annual European Symposium on Algorithms, vol. 5193 of Lecture Notes in Computer Science, 2008, pp. 284–296.

[8] CPLEX, IBM ILOG CPLEX Optimization Studio. http://www-01.ibm.com/software/commerce/optimization/-cplex-optimizer/.

[9] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall, 1999.

[10] P. Eades and L. Xuemin, *How to draw a directed graph*, IEEE Transactions on Visual Languages, (1989), pp. 13–17.

[11] E. R. Gansner, E. Koutsifios, S. C. North, and K. P. Vo, *A technique for drawing directed graphs*, IEEE Transactions on Software Engineering, 19 (1993), pp. 214–230.

[12] M. Garey, D. Johnson, and L. Stockmeyer, *Some simplified np-complete graph problems*, Theoretical Computer Science, 1 (1976), pp. 237 – 267.

[13] M. Jünger and P. Mutzel, *2–Layer Straightline Crossing Minimization: Performance of Exact and Heuristic Algorithms*, Journal of Graph Algorithms and Applications (JGAA), 1 (1997), pp. 1–25.

[14] Kusnadi, J. Carothers, and F. Chow, *Hierarchical graph visualization using neural networks*, IEEE Transactions on Neural Networks, 8 (1997), pp. 794–799.

[15] P. Mutzel, *An alternative method to crossing minimization on hierarchical graphs*, SIAM J. Optimization, 11 (2001), pp. 1065–1080.

[16] J. Stolfi, H. A. D. d. Nascimento, and C. F. X. d. Mendonça, *Heuristics and pedigrees for drawing directed graphs*, Journal of the Brazilian Computer Society, 6 (1999), pp. 38 – 49.

[17] K. Sugiyama, S. Tagawa, and M. Toda, *Methods for visual understanding of hierarchical system structures*, IEEE Transactions on Systems, Man, and Cybernetics, 11 (1981), pp. 109–125.