Theoretical Computer Science 554 (2014) 74–81

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

A linear edge kernel for two-layer crossing minimization

Yasuaki Kobayashi [a,*], Hirokazu Maruta [a], Yusuke Nakae [b], Hisao Tamaki [a]

[a] Department of Computer Science, Meiji University, Higashimita 1-1-1, Tama-ku, Kawasaki, Kanagawa, 214-8571, Japan
[b] Business Information Technical Systems, Taito-ku, Tokyo, Japan

ARTICLE INFO

Article history:
Received 3 September 2013
Received in revised form 20 February 2014
Accepted 4 June 2014
Available online 11 June 2014

Keywords:
Crossing minimization
Fixed parameter tractable
Graph drawing
Kernelization
Two-layer drawing

ABSTRACT

The two-layer crossing minimization problem (TLCM), given a bipartite graph G with n vertices and a positive integer k, asks whether G has a two-layer drawing with at most k crossings. We consider a simple generalization of TLCM called leaf-edge-weighted TLCM (LEW-TLCM), where we allow positive weights on edges incident to leaves, and show that this problem admits a kernel with $O(k)$ edges provided that the given graph is connected. As a straightforward consequence, LEW-TLCM (and hence TLCM) has a fixed parameter algorithm that runs in $2^{O(k \log k)} + n^{O(1)}$ time which improves on the previously best known algorithm with running time $2^{O(k^3)}n$.

1. Introduction

A *two-layer drawing* of a bipartite graph G with bipartition (X, Y) of vertices places vertices in X on one line and those in Y on another line parallel to the first and draws edges as straight line segments between these two lines. We call these parallel lines *layers* of the drawing. A *crossing* in a two-layer drawing is a pair of edges that intersect each other at a point not representing a vertex. Note that the set of crossings in a two-layer drawing of G is completely determined by the order of the vertices in X on one layer and the order of the vertices in Y on the other layer. The problem to find a two-layer drawing whose crossing number is the minimum is called *two-layer crossing minimization*, TLCM for short. This problem is known to be NP-hard [1]. (In fact, the proof of NP-completeness due to [1] is for multigraphs. However, the problem remains NP-complete for simple graphs [2].) There are polynomial time algorithms for trees [3] and bipartite permutation graphs [4]. We consider this problem and its generalization on a parameterized perspective described as follows. An edge is a *leaf edge* if it is incident to a leaf (a vertex of degree one); a *non-leaf edge* otherwise.

Two-Layer Crossing Minimization (TLCM)
Instance: bipartite graph $G = (V(G), E(G))$
Parameter: k
Question: Is there a two-layer drawing of G with at most k crossings?

* Corresponding author. Tel.: +81 44 934 7448.
E-mail address: yasu0207@cs.meiji.ac.jp (Y. Kobayashi).

http://dx.doi.org/10.1016/j.tcs.2014.06.009
0304-3975/

Leaf-Edge-Weighted Two-Layer Crossing Minimization (LEW-TLCM)
Instance: bipartite graph $G = (V(G), E(G))$, function $w : E(G) \to \mathbb{N}$ with $w(e) = 1$ for every non-leaf edge $e \in E(G)$
Parameter: k
Question: Is there a two-layer drawing of G with crossings of total weight at most k, where a crossing (e, e') has weight $w(e)w(e')$?

Clearly, LEW-TLCM is a generalization of TLCM. In this paper, we give kernelizations for these problems.

A *kernelization* [5] for a parameterized problem is an algorithm that, given an instance I and a parameter k, computes an instance I' and a parameter k' of the same problem in time polynomial in k and the size of I such that

1. (I, k) is feasible if and only if (I', k') is feasible,
2. the size of I' is bounded by a computable function f in k, and
3. k' is bounded by a function g in k.

When the functions f and g are polynomial, we call the algorithm a *polynomial kernelization* and its output a *polynomial kernel*.

TLCM is a special case of a problem called h-layer crossing minimization which decides if a given graph has an h-layer drawing with at most k crossings. This problem is fixed parameter tractable [6] when parameterized by $k + h$. The running time of the algorithm of [6] is $2^{O((h+k)^3)}n$. Besides having a large exponent in the running time, this algorithm is rather complicated, involving, in particular, the fixed parameter algorithm for pathwidth due to Bodlaender and Kloks [7,8] and is not easy to implement. It is natural to ask if we can obtain a simpler and faster fixed parameter algorithm for the special case of $h = 2$, namely TLCM. To the best of the present authors' knowledge, neither a faster algorithm for TLCM than the one given in [6] nor its polynomial kernelization is previously known.

In contrast to TLCM, several fixed parameter algorithms are known for *two-layer planarization* (TLP), where the objective is to find a subset of edges of size at most k of the given graph whose removal enables a two-layer drawing without any crossings. This problem is fixed parameter tractable [9] and can be solved in time $O(k \cdot 3.562^k + n)$ where n is the number of vertices of the given graph [10]. Moreover, a kernel with $O(k)$ edges for TLP is known [9].

Another related problem is *one-sided crossing minimization* (OSCM), which asks for a two-layer drawing of the given bipartite graph with minimum number of crossings, but with the vertex order in one layer fixed as part of the input. OSCM is also NP-hard [11]. The fixed parameter tractability of OSCM seems to be better-studied [12–15] than TLCM. More specifically, [12] gives the first fixed parameter algorithm, [13] gives a faster fixed parameter algorithm and a kernel with $O(k^2)$ edges, and [14,15] give subexponential fixed parameter algorithms with running time $2^{O(\sqrt{k}\log k)} + n^{O(1)}$ and $O(3^{\sqrt{2k}} + n)$, respectively.

Our results are as follows.

Theorem 1. *TLCM admits a kernel with $O(k^2)$ edges provided that the given graph is connected.*

Theorem 2. *LEW-TLCM admits a kernel with $O(k)$ edges provided that the given graph is connected.*

The second theorem implies a fixed parameter algorithm with running time $2^{O(k \log k)} + n^{O(1)}$ for both TLCM and LEW-TLCM via the standard approach: given an instance of TLCM or LEW-TLCM, we construct a kernel with $O(k)$ edges in LEW-TLCM for each connected component and then do an exhaustive search to find a solution for each of these kernels. Note that all but k connected components are crossing-free, assuming that the given instance is feasible, which can be detected by a simple linear time algorithm [16].

We remark that, although some of the lemmas needed for the kernelization are non-trivial, the kernelization algorithm itself is quite simple and easy to implement. Moreover, our technique is completely different from those used in the previous results on OSCM and related problems mentioned above.

We sketch here rough ideas behind our method. It is based on the well-known observation that a connected bipartite graph has a two-layer drawing without any crossings if and only if it is a caterpillar, a tree in which every vertex is either on a path (called a spine) or adjacent to a vertex on the spine. We observe that, if a graph has a two-layer drawing with k crossings, then the graph consists of a caterpillar and at most k additional edges. If this spine is long, then it contains many edges that can be contracted without changing the crossing number. Our lemmas enable us to identify those contractable edges and ensure that the graph after contraction has $O(k)$ non-leaf edges.

A preliminary version [17] of this paper showed that there are kernels with at most $10k^2 + 24k + 7$ edges for TLCM and with at most $20k + 7$ edges for LEW-TLCM. In this paper, we improve these bounds to $5k^2 + 10k + 7$ for TLCM and to $10k + 7$ for LEW-TLCM, respectively.

The rest of this paper is organized as follows. In Section 2, we give preliminaries for TLCM. In Section 3, we describe a kernelization for TLCM whose output has $O(k^2)$ edges, proving some lemmas necessary for this kernelization. In Section 4, we show that the same method works for LEW-TLCM and gives a kernelization whose output has $O(k)$ edges. Finally, Section 5 contains the conclusion.