

Fixed Parameter Algorithms for ONE-SIDED CROSSING MINIMIZATION Revisited

Vida Dujmović¹, Henning Fernau^{2,3}, and Michael Kaufmann²

¹ McGill University, School of Computer Science,
3480 University St., Montreal, QC H3A 2A7, Canada,
vida@cs.mcgill.ca

² Universität Tübingen, WSI für Informatik, Sand 13,
72076 Tübingen, Germany
fernau / mk@informatik.uni-tuebingen.de

³ The University of Newcastle, School of Electr. and Computer Science,
University Drive, Callaghan, NSW 2308, Australia
fernau@newcastle.edu.au

Abstract. We exhibit a small problem kernel for the problem ONE-SIDED CROSSING MINIMIZATION which plays an important role in graph drawing algorithms based on the Sugiyama layering approach. Moreover, we improve on the search tree algorithm developed in [5] and derive an $\mathcal{O}(1.4656^k + kn^2)$ algorithm for this problem, where k upperbounds the number of tolerated crossings of straight lines involved in the drawings of an n -vertex graph. Relations of this graph-drawing problem to the algebraic problem of finding a weighted linear extension of an ordering similar to [7] are exhibited.

1 Introduction and Problem Definition

We consider the following problem k -ONE-SIDED CROSSING MINIMIZATION, called k -OSCM for short in this paper: Given: A bipartite graph $G = (V_1, V_2, E)$ and a linear order $<_1$ on V_1 . Parameter: k Question: Is there a linear order $<$ on V_2 such that, when the vertices from V_1 are placed on a line (also called layer) L_1 in the order induced by $<_1$ and the vertices from V_2 are placed on a second layer L_2 (parallel to L_1) in the order induced by $<$, then drawing straight lines for each edge in E will introduce no more than k edge crossings.

We denote by OSCM the optimization version (that is, non-parameterized version) of this problem. (k -)OSCM is the key procedure in the well-known layout framework for layered drawings commonly known as the Sugiyama algorithm. After the first phase (the assignment of the vertices to layers), the order of the vertices within the layers has to be fixed such that the number of the corresponding crossings of the edges between two adjacent layers is minimized. Finally, the concrete position of the vertices within the layers is determined according to the previously computed order.

The crossing minimization step, although most essential in the Sugiyama approach, is an NP-complete problem. The most commonly used method is the

layer-by-layer sweep where, starting from $i = 1$, the order for L_i is fixed and the order for L_{i+1} that minimizes the number of crossings amongst the edges between layers L_i and L_{i+1} is determined.

After increasing index i to the maximum layer index, we turn around and repeat this process from the back with decreasing indices. In each step, an OSCM problem has to be solved. Unfortunately, this seemingly elementary problem is NP-complete [6], even for sparse graphs [8]. Jünger and Mutzel [7] transformed the OSCM problem to a linear ordering problem which they solve with the branch and cut method.

This elementary graph drawing problem attracted several researchers from the area of fixed-parameter tractable (FPT) algorithms [2]. The first approaches to the more general variant of this problem have been published by Dujmović *et al.* in [3] and [4]. The last one has been greatly improved by Dujmović and Whitesides [5] who achieved an $\mathcal{O}(1.6182^k n^2)$ algorithm for this problem.

In this paper, we derive an $\mathcal{O}(1.4656^k + kn^2)$ algorithm, which significantly lowers the constants involved in the exponential search tree algorithm part.

Moreover, we exhibit a small problem kernel for this problem, which has not been done before.

More specifically, this means that, with the aid of some reduction rules, we can arrive at an instance of k -OSCM with the number of vertices involved being bounded by $3k(k + \frac{1}{2})$.

2 Basics

We start with some formalities. The number of vertices of a graph $G = (V, E)$ is denoted by $n = |V|$. For each vertex $v \in V$, let $N(v)$ denote the set of vertices adjacent to v and let $\deg(v) = |N(v)|$ denote the degree of v in G . The subgraph of G induced by the set of vertices $V' \subseteq V$ is denoted by $G[V']$. A bipartite graph $G = (V_1, V_2, E)$ together with linear orderings on V_1 and on V_2 is also called a *drawing* of G . This formulation implicitly assumes a drawing of G where the vertices of V_1 and V_2 are drawn on two (virtual) horizontal lines, the line L_1 corresponding to V_1 being above the line L_2 corresponding to V_2 . If $u < v$ for two vertices on L_1 or on L_2 , we will also say that u is *to the left of* v . A linear order on V_2 that minimizes the number of crossings subject to the fixed linear order of V_1 is called an *optimal ordering* and the corresponding drawing of G is called an *optimal drawing*. Since the positions of isolated vertices in any drawing are irrelevant, we disregard isolated vertices of the input graph G in what follows.

If $|V_2| = 2$, then there are only two different drawings. This gives us the useful notion of a *crossing number*. Given a bipartite graph $G = (V_1, V_2, E)$ with $|V_2| > 1$, for any two distinct vertices $a, b \in V_2$, define c_{ab} to be the number of crossings in the drawing of $G[\{a, b\} \cup (N(a) \cup N(b))]$ when $a < b$ is assumed. Furthermore, for any $a \in V_2$ with $\deg(v) > 0$, let l_a be the leftmost neighbour of a on L_1 , and r_a be the rightmost neighbour of a . We call two vertices $a, b \in V_2$ *interfering* or *unsuited* if there exists some $x \in N(a)$ with $l_b < x < r_b$, or there exists some $x \in N(b)$ with $l_a < x < r_a$. Otherwise, they are called *suited*.

Observe that, for $\{a, b\}$ suited, $c_{ab} \cdot c_{ba} = 0$. Dujmović and Whitesides have shown that, in any optimal ordering $<$ of the vertices of V_2 , we find $a < b$ if $r_a \leq l_b$.

This means that all suited pairs appear in their *natural ordering*.

In the next section we will need the following general notation. A directed graph (digraph) obtained from an undirected graph G by replacing every edge $\{u, v\}$ by the two arcs (u, v) and (v, u) is denoted by $D(G)$. The undirected graph obtained from a digraph G by putting an edge $\{u, v\}$ whenever there is an arc (u, v) in G is denoted by $U(G)$. Finally, the graph complement of G is denoted by G^c .

It is important to note that we only deal with simple (di-)graphs, i.e., graphs having no (self-)loops or multiple edges. The set of arcs of a digraph G is denoted by $A(G)$.

3 Getting a Small Kernel

In this section, we give a drastic reduction of the complexity of k -OSCM. Recall that a partial order is an irreflexive, asymmetric and transitive relation. A partially ordered set (or poset), denoted by $P = (V, A)$, is a set V taken together with a partial order A on it. For two distinct elements $a, b \in V$, if either $a < b$ or $b < a$ in A , then the pair $\{a, b\}$ is *comparable* in P ; else the pair is *incomparable*. A partial order is linear if every pair of elements of V is comparable. If $A \subseteq V \times V$ is a partial order on V , then we call any partial order $L \supseteq A$ a *completion of A* . More algebraically speaking, it can also be called a (weighted) *linear extension*. As is well known, every poset $P(V, A)$ can be represented as a directed acyclic digraph with vertex set V and arc relation A which is a partial order. For simplicity, we will equate posets and directed acyclic graphs in the following, so we refer to posets as *ordered* or *linear digraphs*.

Consider the following problem k -WEIGHTED COMPLETION OF AN ORDERING (k -WCO):

Given: An ordered digraph $P = (V, A)$ and a cost function c mapping $A(D([U(P)]^c))$ into the nonnegative integers; by setting c to zero for arcs in $A(D(U(P)))$, we can interpret the domain of c as $V(P) \times V(P)$. Parameter: k
Question: Is there a selection A' of arcs from $A(D([U(P)]^c))$ such that the transitive closure $(A' \cup A(P))^+$ is a linear order and $\sum \{c(a) \mid a \in (A' \cup A(P))^+\} \leq k$?

It is quite obvious that OSCM can be solved with the help of WCO; the linear order which is aimed at is the permutation of the vertices on the second layer which minimizes the number of crossings involved, so that the crossing number c_{ab} is the cost of the arc (a, b) in the digraph model. Since it is easy to see that WCO is nondeterministically solvable in polynomial time, and since Eades and Wormald [6] have shown that OSCM is NP-hard, we can immediately deduce:

Lemma 1. *WCO is NP-complete.*

Jünger and Mutzel [7] gave an ILP-formulation of the OSCM problem as a linear ordering problem which they could solve with a branch and cut method.

In contrast, our algorithm for OSCM (both the kernelization and the search tree part) can be seen as growing larger and larger parts of the linear order of the vertices of the second layer we are aiming at. When settling the ordering between a and b , we also say that we are *committing* $a < b$ or $b < a$.

Dujmović and Whitesides have shown that, for each instance of OSCM, suited pairs appear in their natural ordering in any optimal drawing. That justifies the following first reduction rule: **RR1**: For every suited pair of vertices $\{a, b\}$ from V_2 with $c_{ba} > 0$, commit $a < b$.

Furthermore, consider a set $S \subseteq V_2$ such that all the vertices in S are adjacent to the exact same set of neighbours in V_1 . It is simple to observe that arbitrarily permuting the vertices of S in any drawing cannot affect the total number of crossings. This observation justifies the following second reduction rule:

RR2: For every pair of vertices $\{a, b\}$ from V_2 with $N(a) = N(b)$, (arbitrarily) commit $a < b$.

Notice that if $c_{ab} = c_{ba} = 0$ then (disregarding isolated vertices) $\deg(a) = \deg(b) = 1$ and $N(a) = N(b)$. Therefore, after applying these two reduction rules to an instance of k -OSCM, we obtain the ordered digraph $P = (V_2, A)$, where a pair of vertices is incomparable in P only if both of the crossing numbers of that pair are nonzero. Hence, we can assume that c is mapping each arc of $A(D([U(P)]^c))$ onto a positive integer. We call the corresponding specialized problem k -POSITIVE COMPLETION OF AN ORDERING (k -PWCO).

Our reasoning up to now shows that this special case is also NP-hard.

Exhaustively applying the following two reduction rules shows that k -PWCO has a small linear-size problem kernel.

RRLO1:

If v is a vertex in ordered digraph $P = (V, A)$ of maximal degree, that is, the sum of the indegrees and outdegrees of v equals $|V| - 1$, then remove v and consider the instance $(P[V - v], k)$.

To see the soundness of this rule, observe that both P and $P[V - v]$ have transitive arc relations. After having applied RRLO1 exhaustively, there is, for each vertex $v \in V$, another vertex $v' \in V(P)$ such that the pair $\{v, v'\}$ is incomparable in P . By the positivity assumption, committing $v < v'$ or $v' < v$ will cost at least one “unit” and will, furthermore, at best maximize the degree of v and v' . By an inductive argument, it follows that the ordered digraph P in a RRLO1-reduced instance of k -PWCO cannot have more than $2k$ vertices.

In an ordered digraph $P = (V, A)$, call two incomparable vertices $u, v \in V$

- *independent with respect to w* if $\{u, w\}$ and $\{v, w\}$ are comparable in P ;
- *dependent with respect to w* if $\{u, w\}$ or $\{v, w\}$ are incomparable in P ;
- *independent (in P)* if $\{u, v\}$ are independent with respect to all $w \in V \setminus \{u, v\}$;
- *dependent (in P)* if $\{u, v\}$ are dependent with respect to some $w \in V \setminus \{u, v\}$;
- *transitive with respect to w* if either $\{u, w\}$ or $\{v, w\}$ is comparable but not both.

Being transitive with respect to w means that committing $v < u$ or $u < v$ commits by transitivity $v < w$, $w < v$, $u < w$ or $w < u$.

RRLO2: If the vertices u and v are independent in a given ordered digraph $P = (V, A)$ and if $c((u, v)) < c((v, u))$, then reduce the problem instance to $((V, A \cup \{(u, v)\}), k - c((u, v)))$.

The soundness of RRLO2 is obvious. In fact, this rule is generalizable:

RRLO q for any fixed $q > 1$: For each connected component $C \subseteq V$ of $[U(P)]^c$ with $|C| \leq q$, solve PWCO optimally on $P[C]$.

The reduced instance will see the orderings between all pairs of vertices from C settled, and the parameter will be reduced accordingly. Note that this implies that all the vertices of C are isolated in the reduced $[U(P)]^c$ and can thus be deleted by the RRLO1. This new rule yields a kernel of size $k \frac{q+1}{q}$, since after exhaustive application of this rule and RRLO1, each connected component of $[U(P)]^c$ has at least $(q+1)$ vertices and thus at least q edges. Correspondingly, at least q arcs with a weight of at least one per arc have to be added per component, therefore there are at most k/q components.

In other words, a kernel size of k can be “approximated” with arbitrary precision. This type of behaviour has not been found in other parameterized algorithms yet. However, this approximation comes at a price: Solving this problem on graphs with q vertices can only be done in exponential time (assuming $P \neq NP$). Still, arbitrary constants q or $q = \log k$ are possible choices for polynomial-time kernelization algorithms. It is therefore desirable to find easy optimality criteria for small values of q (as $q = 2$).

The soundness of the reduction rule RRLO q is readily seen. Assume that C is a connected component of $[U(P)]^c$. Consider a vertex $x \notin C$. Since C is a connected component in $[U(P)]^c$, x is comparable with each $y \in C$ (otherwise, y and x would be connected in the complement graph). Hence, C can be partitioned into $C_\ell = \{y \in C \mid y < x\}$ and $C_r = \{y \in C \mid x < y\}$. Since $C_\ell < x < C_r$, either $C_\ell = \emptyset$ or $C_r = \emptyset$; otherwise, there would be no connection between vertices of C_ℓ and vertices of C_r in the complement graph. Hence, x will never be ordered “in-between” two vertices from C . Therefore, we can conclude:

Theorem 1. *Fix some $1 < \alpha \leq 1.5$. Then, each instance of k -PWCO admits a problem kernel of size αk .*

Considering k -OSCM in a slightly more general fashion, namely, allowing some pairs of V_2 to be already committed as part of the input, we can conclude:

Corollary 1. *Fix some $1 < \alpha \leq 1.5$. Then, each instance of k -OSCM can be reduced to an instance $(P = (V_2, A), k)$ of k -PWCO, with $|V_2| \leq \alpha k$.*

Proof. Applying RRLO1 and RRLO2 (and RRLO q) to k -OSCM exhaustively results in a partial order on V_2 where the crossing numbers of the incomparable pairs are positive. Therefore, this partial order on V_2 , together with the positive crossing numbers of incomparable pairs comprises an instance of k -PWCO, which by Theorem 1 has the kernel of size $|V_2| \leq \alpha k$. \square

This has not yet given us a problem kernel for the original problem, since $|V_1|$ (and hence $|E|$) has not yet been bounded by a function of k .

With that aim, consider the following simple reduction rule

RRlarge: If $c_{ab} > k$ then do: if $c_{ba} \leq k$ then commit $b < a$ else return NO.

This is clearly a sound rule. Moreover notice, that if a vertex $a \in V_2$ has $\deg(a) > 2k + 1$, then for every vertex $b \in V_2$, $c_{ab} > k$ or $c_{ba} > k$ are true. Therefore, after performing RRlarge exhaustively in combination with RRLO1, all the vertices of V_2 of degree larger than $2k + 1$ will have been removed from the OSCM-instance. This yields:

Theorem 2. *For some $1 < \alpha \leq 1.5$, k -OSCM admits a problem kernel $G = (V_1, V_2, E)$ with $|V_1| \leq (2k + 1)|V_2| (\leq 3k(k + \frac{1}{2}))$, $|V_2| \leq \alpha k$, and $|E| \leq (2k + 1)|V_2|$.*

The following algorithm summarizes how to obtain a problem kernel:

Algorithm 1 (Kernelization for k -OSCM).

Compute the crossing numbers c_{ab} for all pairs (a, b) .

If $k < \sum_{a, b \in V_2} \min(c_{ab}, c_{ba})$ then answer NO.

If $k \geq \sum_{a, b \in V_2} \max(c_{ab}, c_{ba})$ then YES; output an arbitrary linear order.

Apply reduction rules RR1, RR2 (and RR3).

Apply reduction rules RRL01, RRL02, and RRlarge exhaustively.

The rule RR3 is derived in Section 6, as it is needed by the search tree part of our algorithm. For now, while considering kernelization only, we can disregard this reduction rule. Since computing the crossing numbers is the predominant computational part (taking time $\mathcal{O}(kn^2)$ even when combined with RRlarge according to Dujmović and Whitesides), we conclude that the kernelization pre-processing takes time $\mathcal{O}(kn^2)$. As a result of this kernelization, the vertices of V_2 are partially ordered. Our search tree algorithm presented in the next section can cope with such “pre-set” instances of k -OSCM. Therefore, we will obtain running time of the form $\mathcal{O}(f(k) + kn^2)$ for the k -OSCM problem.

4 A General Overview of the Search Tree Algorithm

Our search tree for the OSCM problem slightly deviates from the one constructed by Dujmović and Whitesides in their paper. Hence, we give some details in what follows. Consider two distinct vertices $a, b \in V_2$ with $c_{ab} = i$ and $c_{ba} = j$. Then, the pair $\{a, b\}$ is also called an i/j pattern.

As is the standard practice when developing search tree based FPT algorithms, each node of a search tree is associated with a problem instance. In the case of k -OSCM, we let an instance consist of an ordered digraph $P = (V_2, A)$ and an integer k' . P contains the arcs corresponding to all the pairs of vertices of V_2 committed thus far, and k' gives the remaining number of allowed edge crossings, that is, $k' = k - \sum_{(v,u) \in A} c_{vu}$. Initially (in the root node) the arcs

of P are determined by the kernelization. In each node of the search tree, if k' is negative, then the node returns NO, otherwise we look for a pair of dependent vertices $\{a, b\}$ that form an i/j pattern such that either $i + j \geq 4$, or $i = 2, j = 1$ and $\{a, b\}$ is transitive. If such a pair $\{a, b\}$ is found we branch the problem instance into two new problem instances. In one we commit $a < b$ and in the other we commit $b > a$; consequently, in each problem instance we update the ordered digraph P and lower the parameter k' accordingly. (By adding (v, w) to A and then computing transitive closure $(A \cup (v, w))^+$ we get the updated ordered digraph $P = (V_2, (A \cup (v, w))^+)$.) If no pair as described above is found, then we commit all the remaining incomparable pairs in P deterministically. The details and the correctness of this approach will be discussed in Section 7.

The running time of FPT algorithms is dominated by the part exponential in parameter k . In our case, that part is bounded by the size of the search tree.

We now analyse that size. Firstly, observe that each internal node of our search tree has two branches. If one branch lowers the parameter k' by b_1 , and the other by b_2 , we denote the corresponding *branching vector* by (b_1, b_2) . Since all i/j patterns with $i = 0$ or $j = 0$ are already committed (by the reduction rules RR1 and RR2 during the kernelization), each internal node of the search tree has $b_1 > 0$ and $b_2 > 0$. Furthermore, each node that branches on a transitive 2/1 pattern commits by transitivity an extra i/j pattern where $i, j > 0$. Thus each internal node has $b_1 + b_2 \geq 4$. Therefore, in the worst case, each node of our search tree has a branching vector $(2, 2)$ or $(3, 1)$. The size $s(k)$ of the search tree obeying these branching vectors can be deduced as follows:

- The recurrence corresponding to the $(2, 2)$ branching vector is $s(k) = 2s(k - 2) + \mathcal{O}(1)$. Solving this recurrence gives $s(k) < 1.4143^k$.
- The recurrence corresponding to the $(1, 3)$ branching vector is $s(k) = s(k - 1) + s(k - 3) + \mathcal{O}(1)$. Solving this recurrence gives $s(k) < 1.4656^k$.

Thus in the worst case, the size of our search tree is less than 1.4656^k . This is in contrast to the search tree with a $(2, 1)$ branching vector derived by Dujmović and Whitesides that gives $s(k) < 1.6181^k$.

In a node of our search tree that does not branch (that is, in a leaf), all the incomparable pairs form 1/1 and 2/1 patterns. To be able to commit these pairs deterministically, we study the structural properties of such patterns in the next section. In Section 7, we finally give a detailed algorithm for the k -OSCM problem and the proof of its correctness.

5 Some Structural Properties of 2/1 and 1/1 Patterns

We start this section with the following known and useful equality [1, Chapter 9]:

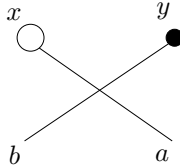
$$c_{ab} + c_{ba} = \deg(a) \deg(b) - |N(a) \cap N(b)|. \tag{1}$$

This implies that if $\deg(b) \leq \deg(a)$ then

$$(\deg(a) - 1) \deg(b) \leq c_{ab} + c_{ba} \leq \deg(a) \deg(b). \tag{2}$$

We now study 1/1 and 2/1 patterns. As a convention, we will label vertices from the first layer by (decorated) letters x, y, z and vertices from the second layer by letters a, b, c . For clarity, we will draw neighbours of a as non-filled circles and neighbours of b as filled-in circles.

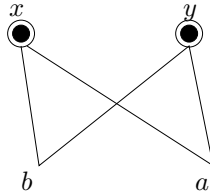
Lemma 2. *In the case of 1/1 or 2/1 patterns, and if $c_{ba} = 1$, we must find the situation depicted in the figure below.*



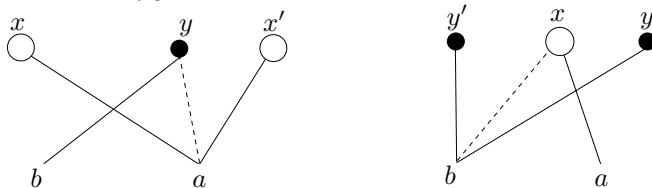
Moreover, each remaining neighbour of a (if any) must be to the right of y (or y itself), while each remaining neighbour of b (if any) must be to the left of x (or x itself). (Otherwise, $c_{ba} > 1$.)

Theorem 3. *If $c_{ab} = c_{ba} = 1$, then there are two basically different situations (that can be obtained by enhancing the situation sketched in Lemma 2):*

1. a and b are each adjacent to x and y only. In other words, $\deg(a) = \deg(b) = 2$ and $N(a) = N(b)$, as illustrated in the figure below.



2. Two subcases arise: (a) If $\deg(b) = 1$, then a has (besides x) another neighbour x' to the right of y . In addition to x and x' , a may only be adjacent to y . (b) If $\deg(a) = 1$, then b has (besides y) another neighbour y' to the left of x . In addition to y and y' , b may only be adjacent to x . Both situations are illustrated in the figure below.



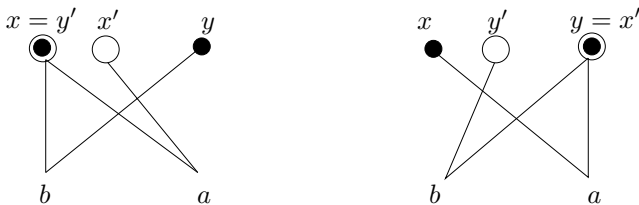
Proof. By inequality (2) either $\deg(a) = \deg(b) = 2$, or one of a, b has degree one. Consider first the case where $\deg(a) = \deg(b) = 2$.

Then by equality (1), $N(a) = N(b)$ as otherwise $c_{ab} + c_{ba} \geq 3$. Consider now the case where $\deg(b) = 1$. Then a has (besides x) exactly one neighbour x' strictly to the right of y as otherwise either the pair $\{a, b\}$ is suited or $c_{ab} > 1$.

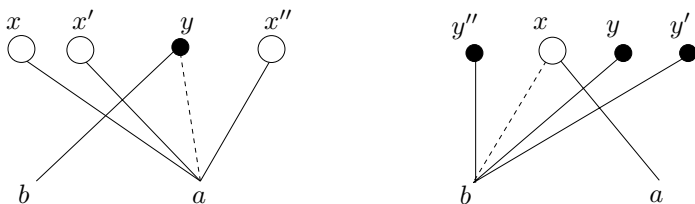
Vertex a cannot have a neighbour strictly to the left of y as otherwise $c_{ab} > 1$. Thus in addition to x and x' , a can only be adjacent to y . The remaining case where $\deg(a) = 1$ is symmetric. □

Theorem 4. *Let the pair $\{a, b\}$ form a 1/2 pattern, that is, $c_{ab} = 1$ and $c_{ba} = 2$. Then there are two basically different situations:*

1. *In the following figures, neither a nor b can have any other neighbours.*



2. *In the following figure (on the left-hand side), in addition to x , x' and x'' , vertex a may only be adjacent to y , while b has no other neighbours. The figure on the right-hand side can be symmetrically interpreted.*



Proof. By inequality (2) either $\deg(a) = \deg(b) = 2$, or one of a, b has degree one. Consider first the case where $\deg(a) = \deg(b) = 2$ and let x' and y' denote the remaining neighbours of a and b , respectively. Having both $x' \neq y$ and $y' \neq x$ is not possible as otherwise by equality (1), $c_{ab} + c_{ba} = 4$. Having both $x' = y$ and $y' = x$ gives the first case of the previous theorem.

Therefore, if $x' = y$ then $x < y' < y$ as otherwise, $\{a, b\}$ is suited or $c_{ba} = 1$. On the other hand, if $y' = x$, then $x < x' < y$ as otherwise, $\{a, b\}$ is suited or $c_{ba} = 1$.

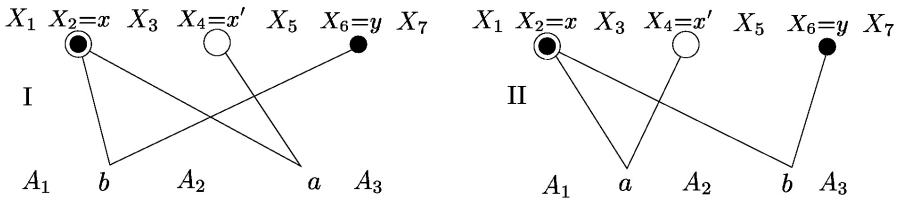
Consider now the case where $\deg(b) = 1$. The case where $\deg(a) = 1$ is symmetric. In addition to x , a has exactly one neighbour x' strictly to the left of y as otherwise, $c_{ba} \neq 2$. Vertex a has exactly one neighbour x'' strictly to the right of y as otherwise, either the pair $\{a, b\}$ is suited or $c_{ab} > 1$. Thus in addition to x , x' and x'' , a can only be adjacent to y . □

6 A Reduction Rule for a 2/1 Pattern

The reduction rule RR2 presented in Section 3 resolves the situation described in the first case of Theorem 3. Unfortunately, the second case of Theorem 3 does not admit a similar simple resolution. Let us now reconsider the 2/1 patterns identified in Theorem 4. The second kind of the pattern described in that theorem is in a sense similar to (but more complicated than) the second pattern of Theorem 3 and hence admits no deterministic solution.

Consider the first case in Theorem 4 with a having a neighbour distinct from x and y . Let I denote any drawing where $b < a$ and let II denote the drawing obtained from drawing I by swapping a and b . Let the total number of crossings in I and II be denoted by c_I and c_{II} respectively.

Consider the following figure and tables.



T_I	A_1	A_2	A_3		T_{II}	A_1	A_2	A_3
X_1	0	2	4		X_1	0	2	4
X_2	0	1	2		X_2	0	1	2
X_3	2	2	2		X_3	2	2	2
X_4	2	2	1		X_4	2	1	1
X_5	3	3	1		X_5	3	1	1
X_6	3	2	0		X_6	3	1	0
X_7	4	2	0		X_7	4	2	0

These tables read as follows: if there are m_{ij} edges connecting vertices from X_i with vertices from A_j , then there are $T_x(X_i, A_j)m_{ij}$ crossings between these m_{ij} edges and the edges shown in the above sketches for case $x \in \{I, II\}$. It is clear that the columns labelled A_1 and A_3 are identical in both tables: swapping a and b can only affect the relative order of vertices in A_2 compared to a and b . The boxed entries show the only differences between the tables. It follows that

$$c_I - c_{II} = c_{ba} - c_{ab} + m_{42} + 2m_{52} + m_{62} > 0,$$

since $m_{42} + 2m_{52} + m_{62} \geq 0$ and $c_{ba} - c_{ab} = 1$. This implies that whenever this situation from Theorem 4 arises, any optimal drawing shows $a < b$.

The first case in Theorem 4 where b has a neighbour distinct from x and y is symmetric; the same analysis applies. This justifies the next reduction rule:

RR3: In the situations described in the first case of Theorem 4, always let $a < b$.

Since in every optimal drawing all the pairs of this type appear in their “cheaper ordering”, we apply this reduction rule at the very beginning of our algorithm, that is we add this rule to the kernelization as shown in Algor. 1.

Having this reduction rule is instrumental in bounding the size of the search tree, as will become clear from the two lemmas in the next section.

7 The Algorithm for the OSCM Problem

We are now ready to present the parameterized algorithm we suggest for OSCM:

Algorithm 2 (A parameterized algorithm for OSCM).

- kernelize

In a node of a search tree with instance (P, k') do

0: if $k' < 0$ return NO.

1: Apply RRL01, RRL02 and RRlarge exhaustively.

2: if in P there is an incomparable i/j pattern $\{a, b\}$ such that $i + j \geq 4$ branch on $a < b$ and $b < a$; update P and k' accordingly

3: else if in P there is a dependent $2/1$ pattern $\{a, b\}$ branch on $a < b$ and $b < a$; update P and k' accordingly

4: else

//the remaining $2/1$ patterns are independent; thus, RRL02 applies
resolve all remaining $1/1$ patterns arbitrarily
update P and k' accordingly; if $k' < 0$ return NO else YES.

The correctness of this algorithm is clear from the analysis presented in the previous sections. What remains is to analyse the branching vectors of the internal nodes in the search tree associated with the algorithm. As required by the analysis presented in Section 3, we now show that the branching vector (b_1, b_2) in each internal node has $b_1 + b_2 \geq 4$ and $b_1, b_2 > 0$.

Lemma 3 (Main Lemma). *Let $\{a, b\}$ be a pair of dependent vertices forming $2/1$ pattern in step 3 of the algorithm. Then $\{a, b\}$ is a transitive pair.*

Proof. Since $\{a, b\}$ is dependent in P , there must be a vertex c such that $\{a, c\}$ or $\{b, c\}$ are incomparable in P . It suffices to show that one of these two pairs are comparable in P . In the step 3 of the algorithm, the only remaining incomparable pairs are $1/1$ patterns of the second type in Theorem 3 and $2/1$ patterns of the second type in Theorem 4. Therefore, let without loss of generality $\deg(a) = 3$ (or $\deg(a) = 4$) and $\deg(b) = 1$. If $\deg(c) \geq 2$ then by the inequality (2), $c_{ac} + c_{ca} \geq 4$ and thus the ordering of $\{a, c\}$ is settled either by RR1 or by the step 2 of the algorithm. Therefore, $\deg(c) = 1$. In that case, the pair $\{b, c\}$ forms a $0/i$ pattern and its ordering is settled by either RR1 or RR2. \square

A direct consequence of this lemma is the following.

Lemma 4. *The branching vector (b_1, b_2) in each internal node of the search tree has $b_1 + b_2 \geq 4$ and $b_1, b_2 > 0$.*

Proof. Based on the reduction rules RR1 and RR2, in each node of a search tree all the incomparable (and dependent) patterns i/j have $i > 0$ and $j > 0$, thus in each node $b_1, b_2 > 0$. Furthermore, all the nodes branched in the step 2, have $i + j \geq 4$ and thus have $b_1 + b_2 \geq 4$. The remaining nodes are branched in step 3. Each such node branches on a dependent 2/1 pattern $\{a, b\}$. By the previous lemma, committing either $a < b$ or $b < a$ determines, without loss of generality, the ordering of a pair $\{a, c\}$. Having been incomparable at step 3 initially, the pair $\{a, c\}$ has $c_{ac}, c_{ca} \geq 1$. Therefore, we have $b_1 + b_2 \geq 4$ in this case, too. \square

8 Conclusions

In this paper, we present a new search tree based parameterized algorithm for the k -OSCM problem. Due to possible rekernelizations [9], we may deduce:

Theorem 5. *The problem of k -OSCM can be solved in time $\mathcal{O}(1.4656^k + kn^2)$.*

It remains to be determined whether further progress is possible, especially in further lowering the base of the exponent in the running time of the search tree algorithm. Our present case analysis shows at two places a branching behaviour which matches the upper bound stated in the previous theorem, namely when branching at 3/1 and at 2/1 patterns. A detailed analysis of 3/1 patterns may be worthwhile, but will probably be very tedious, requiring considerations of all the possible combinations.

Let us finally remark that even with the possibly more difficult problem k -PWCO, we are able to derive a search tree algorithm with a better branching behaviour than Dujmović and Whitesides did for k -OSCM. Since the result is of certain interest on its own, we will state it in the following.

Theorem 6. *k -PWCO can be solved in time $\mathcal{O}(1.5175^k + kn^2)$.*

In fact, the worst case branching vector in our algorithm is $(2, 4, 4, 4)$.

Acknowledgments. We are grateful for hints of F. Rosamond, P. Shaw and M. Suderman on draft versions of this paper.

References

1. Di Battista, G., Eades P., Tamassia R. and I. G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall, 1999.
2. R. Downey and M. Fellows. *Parameterized Complexity*, Springer, 1999.
3. V. Dujmović, M. Fellows, M. Hallet, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosemand, M. Suderman, S. Whitesides, and D. Wood, An efficient fixed parameter approach to two-layer planarization. In P. Mutzel and M. Jünger, eds., *Graph Drawing GD'01*, LNCS 2265, pages 1-15. Springer, 2001.
4. V. Dujmović, M. Fellows, M. Hallet, M. Kitching, G. Liotta, C. McCartin, N. Nishimura, P. Ragde, F. Rosemand, M. Suderman, S. Whitesides, and D. Wood, On the parameterized complexity of layered graph drawing, In F. Meyer auf der Heide, ed., *European Symposium on Algorithms ESA '01*, LNCS 2161, pages 488-499. Springer, 2001.

5. V. Dujmović and S. Whitesides. An efficient fixed parameter tractable algorithm for 1-sided crossing minimization. In M. T. Goodrich and S. G. Kobourov, eds., *Graph Drawing GD'02*, LNCS 2528, pages 118–129. Springer, 2002.
6. P. Eades and N. C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11:379–403, 1994.
7. M. Jünger and P. Mutzel. 2-layer straightline crossing minimization: performance of exact and heuristic algorithms. *J. Graph Algorithms Appl.*, 1:1–25, 1997.
8. X. Munoz, W. Unger, and I. Vrto, One sided crossing minimization is NP-hard for sparse graphs. In P. Mutzel and M. Jünger, eds., *Graph Drawing GD'01*, LNCS 2265, pages 115–123. Springer, 2001.
9. R. Niedermeier and P. Rossmanith. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73:125–129, 2000.
10. Sugiyama K., S. Tagawa and M. Toda. Methods for visual understanding of hierarchical system structures, *IEEE Transactions on Systems, Man and Cybernetics*, 11:109-125, 1981.