Layered Drawings of Graphs with Crossing Constraints*

Irene Finocchi

Department of Computer Science
University of Rome "La Sapienza"
finocchi@dsi.uniroma1.it
http://www.dsi.uniroma1.it/~finocchi

Abstract. We study the problem of producing hierarchical drawings of layered graphs when some pairs of edges are not allowed to cross. We show that deciding on the existence of a drawing satisfying at least k constraints from a given set of non-crossing constraints is NP-complete even if the graph is 2-layered and even when the permutation of the vertices on one side of the bipartition is fixed. We also propose simple constant-ratio approximation algorithms for the optimization version of the problem and we discuss how to extend the well-known hierarchical approach for creating layered drawings of directed graphs with the capability of minimizing the number of edge crossings while maximizing the number of satisfied non-crossing constraints.

1 Introduction

The problem of embedding graphs in the plane when only some pairs of edges are allowed to cross, known in literature as *realizability*, has been formally introduced in [12]. It finds application, e.g., in VLSI layout, where the crossings between certain pairs of edges must be avoided due to the physical realization of connectors, and in constrained graph layout, where users can specify requirements on the visualization that should be fulfilled by the drawing algorithm. In these settings, it is fundamental to decide if an embedding of a graph satisfying a given set of non-crossing constraints exists, and, if the answer is positive, to find it. Realizability has been proved to be NP-hard in [10], but surprisingly it is not known to belong to the class NP, turning out to be a very interesting problem from a theoretical perspective. Its theoretical relevance is mostly due to the connection with the classical themes of planarity and crossing numbers and with the recognition of string graphs, i.e., intersection graphs of curves in the plane. The relation between realizability and string graphs is addressed in [10,11]; here we limit to point out that no finite algorithm for the recognition of string graphs is known and that, given a graph G, it is possible to build an instance of realizability which can be satisfied if and only if G is a string graph [11]. It is also worth

* Work supported in part by the project "Algorithms for Large Data Sets: Science and Engineering" of the Italian Ministry of University and of Scientific and Technological Research (MURST 40%).

J. Wang (Ed.): COCOON 2001, LNCS 2108, pp. 357–368, 2001.

observing that realizability is a generalization of the well-known planarity problem, which requires to decide if a graph is planar and clearly coincides with the special case where no pair of edges is allowed to cross. We recall that planarity can be checked in linear time [9].

This paper is concerned with the realizability problem in the special case of hierarchical drawings [3], i.e., drawings where vertices are constrained to lie on a set of parallel lines and edges are represented as polygonal chains. The hierarchical drawing convention is widely used to represent procedure call dependencies, class hierarchies, is-a relationships, and several economic and social structures [3,5,14]. In the remainder of the paper we will refer to the vertex set on each line as *layer* or *level*, and sometimes to the drawing itself as *layered drawing*. Without loss of generality we also assume to deal with *proper* layered graphs, i.e., layered graphs whose edges span only consecutive levels: it is always possible to reduce to this case by properly adding dummy vertices to split edges with end-points in non-consecutive layers. Under this hypothesis each edge in a hierarchical drawing is simply represented by a straight-line. The main question that we address is the following: *Given a proper layered graph G, a set of edge pairs C, and an integer $k > 0$, does a hierarchical drawing of G exist s.t. at least k edge pairs in C do not cross?* We refer to this problem as *hierarchical realizability* (HR) and to the elements of C as *non-crossing constraints*. We study the computational complexity of some variants of hierarchical realizability and we describe approximation strategies for the optimization versions of these problems, that require to find a maximum realizable subset of C. We also show how algorithms for the variants that we consider can be used as subroutines to boost existing graph drawing algorithms to support non-crossing constraints. Our results can be easily generalized to constraints concerning the relative positions of vertices within each layer, though for simplicity we will not consider them in this paper.

In more detail, the presentation of our results is organized as follows. In Section 2 we prove that deciding on the realizability in case of hierarchical drawings is not easier than deciding on the realizability for generic drawings: we show that HR, which clearly belongs to NP, remains NP-hard even if G is 2-layered and $k = |C|$. It is interesting to observe that the strictly related problem of deciding if a layered graph has a hierarchical drawing *without* edge crossings can be solved in linear time [8]. From an optimization point of view, it is trivial to approximate a maximum cardinality subset of realizable constraints with expected ratio 2, yet we can exhibit instances for which this bound is tight. In addition to the general case, in Section 3 we consider a one-sided realizability problem where G is 2-layered and the vertex ordering of a layer is fixed; we show this problem to be NP-hard for a generic k, approximable with constant ratio 2, and polynomial if $k = |C|$. Based on the one-sided formulation, in Section 4 we finally address the constrained edge crossing minimization problem in layered drawings, i.e., the problem of drawing hierarchically a graph with the minimum number of edge crossing while respecting a set of non-crossing constraints. We consider the well-known Sugiyama's approach for producing layered drawings of directed

graphs [14] and we extend it in order to deal with non-crossing constraints. In particular, we provide a mechanism for identifying a subset $C' \subseteq C$ of realizable constraints, and then we show how to minimize edge crossings while satisfying all the constraints in C'.

2 Hierarchical Realizability

In this section we prove that deciding about the hierarchical realizability of a proper layered graph under a given set of non-crossing constraints is NP-complete. From an optimization point of view, we show that a maximum cardinality subset of realizable constraints can be easily approximated with expected ratio 2 and we exhibit instances for which this bound is tight.

Lemma 1. *HR* $\in NP$.

Proof. Since edge crossings in layered drawings depend only on the permutation of the vertices within each layer, a non-deterministic algorithm has simply to guess an ordering of the vertices of each layer and to check in polynomial time the realization of at least k constraints.

In the following we show the NP-hardness of HR on two-layered (i.e., bipartite) graphs. For brevity, we call bipartite straight-line drawing a hierarchical drawing with only two layers. Let $B = (V_0, V_1, E)$ be a bipartite graph with $n = n_0 + n_1$ vertices and m edges. Without loss of generality, we assume that for any $(a, b) \in E$ $a \in V_0$ and $b \in V_1$. We also denote by π_0 and π_1 two permutations of the vertices in V_0 and V_1, respectively. Let $C \subseteq \binom{E}{2}$ be a set of pairs of edges. The hierarchical realizability problem on bipartite graphs consists of asking for a set $C' \subseteq C$, $|C'| \geq k$, and for a bipartite straight-line drawing of B such that for any $(a, b; c, d) \in C'$ edges (a, b) and (c, d) do not cross in the drawing. More formally, the problem can be stated as follows:

Bipartite realizability (in short, BR): Let $B = (V_0, V_1, E)$ be a bipartite graph, let C be a set of non-crossing constraints, and let k be a positive integer. Do two permutations π_0 and π_1 exist such that the bipartite straight-line drawing of B induced by π_0 and π_1 satisfies at least k constraints?

In the special case where the permutation of the vertices in a layer (e.g., V_0) is fixed, we have the *one-sided bipartite realizability* problem (in short, OBR), whose complexity and approximability are discussed in Section 3.

Before proving the NP-completeness of BR, we state it in a slightly different form which is more convenient for our purposes. Indeed, it is easy to see that asking for two permutations satisfying a set of non-crossing constraints C' is equivalent to asking for two bijective functions π_0 and π_1 such that:

$$\forall (a, x; b, y) \in C' \quad (\pi_0(a) - \pi_0(b)) \cdot (\pi_1(x) - \pi_1(y)) \geq 0 \tag{1}$$

Based on this observation, we prove the NP-hardness of BR by means of a polynomial time reduction from a total ordering problem arising in the computational biology field, called *betweenness*, which has been shown to be NP-complete in [13] and is defined as follows:

Betweenness: Given a finite set S of real variables and a set of triples $T \subseteq S^3$, does an ordering of the variables exist such that for each $(a, b, c) \in T$ variable b is positioned between variables a and c in the ordering?

We refer to the triples in T as *ordering constraints*. A solution for betweenness is a bijective function $f : S \to \{1..|S|\}$ such that:

$$\forall (a, b, c) \in T \quad (f(a) - f(b)) \cdot (f(b) - f(c)) \geq 0 \tag{2}$$

We remark that, in spite of the apparent similarity of Equation 1 and Equation 2, the former involves four distinct vertices and two different functions, while the latter only three variables and a single function. A one-to-one correspondence is therefore not straightforward.

Lemma 2. *Bipartite realizability is NP-hard.*

Proof. Moving from an instance of betweenness, we build an instance of BR as follows. The bipartite graph $B = (V_0, V_1, E)$ is such that: (a) $V_0 = S$; (b) V_1 contains a distinct vertex x' for each item $x \in S$, i.e., $V_1 = \{x' : x \in S\}$; (c) $E = \{(x, x') : x \in S\} \cup \{(a, b'), (b, c') : (a, b, c) \in T\}$. The set of non-crossing constraints C is the union of two disjoint sets C_1 and C_2, where $C_1 = \{(x, x'; y, y') : x, y \in S,\ x \neq y\}$ and $C_2 = \{(a, b'; b, c') : (a, b, c) \in T\}$. Note $|C_1| = |S|(|S| - 1)$ and $|C_2| = |T|$. $k = |C_1| + |C_2|$, i.e., the problem is to decide if *all* the non-crossing constraints can be realized.

It is easy to verify that the reduction above requires $O(|T| + |S|^2)$ time. To prove that k constraints in C can be realized iff the instance of betweenness admits a solution we exploit the one-to-one correspondence between ordering constraints in T and crossing constraints in C_2 and the fact that realizing all the constraints in C_1 forces $\pi_0(a) = \pi_1(a')$ for any pair of corresponding vertices $a \in V_0$ and $a' \in V_1$.

Let us first suppose that two permutations π_0 and π_1 exist able to satisfy all the constraints in C. Then $f = \pi_0$ represents a solution for betweenness. Indeed, let $(a, b, c,) \in T$ be an ordering constraint in the instance of betweenness. Let us consider the constraints $(b, b'; c, c') \in C_1$ and $(a, b'; b, c') \in C_2$: these constraints necessarily exist by construction. Since π_0 and π_1 satisfy all the non-crossing constraints, by Equation 1 we know that: $(\pi_0(a) - \pi_0(b)) \cdot (\pi_1(b') - \pi_1(c')) \geq 0$ and $(\pi_0(b) - \pi_0(c)) \cdot (\pi_1(b') - \pi_1(c')) \geq 0$ By multiplying the left members of these inequalities, we obtain $(\pi_0(a) - \pi_0(b)) \cdot (\pi_0(b) - \pi_0(c)) \geq 0$, proving that the ordering constraint $(a, b, c) \in T$ is satisfied by $f = \pi_0$.

Conversely, let us assume that the instance of betweenness admits a solution f and let us define $\pi_0 = f$ and $\forall a' \in V_1\ \pi_1(a') = f(a)$. Note that $\forall a \in S$ $\pi_0(a) = \pi_1(a')$. Any constraint $(a, a'; b, b') \in C_1$ is trivially satisfied by π_0 and π_1, since $\pi_0(a) - \pi_0(b) = \pi_1(a') - \pi_1(b')$ and therefore their product is ≥ 0. Let $(a, b'; b, c')$ be any constraint in C_2. As above, $(\pi_0(b) - \pi_0(c)) \cdot (\pi_1(b') - \pi_1(c')) \geq 0$. Moreover, since $\pi_0 = f$ satisfies constraint $(a, b, c) \in T$, by Equation 2 $(\pi_0(a) - \pi_0(b)) \cdot (\pi_0(b) - \pi_0(c)) \geq 0$. By multiplying the left members of the two inequalities we finally get $(\pi_0(a) - \pi_0(b)) \cdot (\pi_1(b') - \pi_1(c')) \geq 0$, i.e., constraint $(a, b'; b, c')$ is satisfied by π_0 and π_1, as well.

The NP-completeness of HR immediately follows from Lemma 1 and Lemma 2:

Theorem 1. *Hierarchical realizability is NP-complete.*

We now consider the problem of approximating a maximum set of realizable constraints. If we choose, for each layer of G, a random ordering of the vertices in that layer, we approximate a maximum set of realizable constraints (chosen from C) with expected ratio 2.

If $c = |C|$ and $c_i = (a, x; b, y)$, $1 \leq i \leq c$, is a non-crossing constraint involving vertices from any two consecutive layers L_j and L_{j+1}, four orderings among $a, b \in L_j$ and $x, y \in L_{j+1}$ are possible, yet only two of them realize c_i. Hence, the probability p_i of having constraint c_i satisfied is equal to $\frac{1}{2}$. Let X_i be a random variable equal to 1 if constraint c_i is realized, 0 otherwise. The quantity $\sum_{i=1}^{m} X_i$ represents the number of realized constraints and the expected value for it is then equal to $E[\sum_{i=1}^{c} X_i] = \sum_{i=1}^{c} E[X_i] = \sum_{i=1}^{c} p_i = \frac{c}{2}$.

We now show that the ratio above is tight, i.e., no algorithm can do better than trying to satisfy half of the constraints in C, unless being able to recognize non-realizable instances of HR. The following lemma is useful at this aim:

Lemma 3. *The number of non-crossing pairs of non-incident edges in any bipartite straight-line drawing of $K_{n,n}$ is equal to $\left(\frac{n(n-1)}{2}\right)^2$.*

Proof. Any bipartite straight-line drawing of $K_{n,n}$ has the same number x of non-crossing pairs of non-incident edges. Then:

$$x = \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\sum_{k<i} \sum_{h<j} 1 + \sum_{k>i} \sum_{h>j} 1 \right) = \left(\tfrac{n(n-1)}{2}\right)^2$$

Let $G = K_{n,n}$ and let C contain a constraint for any pair of non-incident edges. It is easy to see that $c = |C| = \frac{n^2(n-1)^2}{2}$ and therefore, due to Lemma 3, the number of realized constraints is always equal to $\frac{c}{2}$ independently of the permutation of the two layers.

3 One-Sided Bipartite Realizability

In this section we focus on the one-sided bipartite realizability problem, that turns out to be very useful in the design of algorithms for solving HR, as we will show in Section 4. Unfortunately, we can prove that it is NP-complete, yet easier than BR, being polynomially solvable if we are interested in the existence of a solution satisfying *all* the constraints in C (i.e., for $k = |C|$).

NP-completeness. For simplicity we prove the NP-hardness of one-sided bipartite realizability in two steps, making use of a simple generalization of this problem which takes into account also *crossing constraints* in addition to the non-crossing ones. We refer to the generalized problem as *one-sided bipartite realizability with mixed constraints* (in short, MBR). MBR is stated exactly as OBR, except for taking into account a set of crossing constraints $I \subseteq \binom{E}{2}$ with

the following meaning: for any $(a, b; c, d) \in I$, edges (a, b) and (c, d) should cross in the bipartite drawing of graph B.

With arguments similar to those used in the proof of Lemma 1, it is easy to see that both OBR and MBR belong to NP. We now prove the NP-hardness of MBR by means of a polynomial-time reduction from the acyclic subgraph problem. Then we further reduce MBR to our original realizability problem OBR. We recall that the acyclic subgraph problem is stated as follows:

Acyclic subgraph (in short, AS): given a directed graph $G = (V, A)$ and a positive integer h, does a set $A' \subseteq A$ exist such that $|A'| \geq h$ and the graph $G' = (V, A')$ is acyclic?

This problem has been proved to be NP-complete in [6].

Lemma 4. *One-sided bipartite realizability with mixed constraints is NP-hard.*

Proof. Given an instance of AS, we build an instance of MBR as follows. The bipartite graph $B = (V_0, V_1, E)$ is such that: (a) $V_1 = V$; (b) V_0 contains a distinct vertex x' for each vertex $x \in V$, i.e., $V_0 = \{x' \; s.t. \; x \in V\}$; (c) $\forall (x, y) \in A$ both (x', y) and (y', x) belong to E. Permutation π_0 is any ordering of the vertices in V_0. For each arc $(x, y) \in A$ we set up a constraint $(x', y; y', x)$: if $\pi_0(x') < \pi_0(y')$ we add $(x', y; y', x)$ to I, otherwise we add $(x', y; y', x)$ to C. $k = h$. Figure 1 provides an example of this reduction. A directed graph G is given in Figure 1(a) and the bipartite graph B obtained from it is shown in Figure 1(b), together with the whole set of constraints obtained choosing $\pi_0 = \langle d', a', c', b' \rangle$.

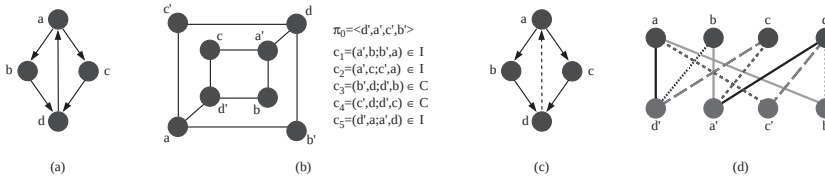

Fig. 1. From acyclic subgraph to one-sided realizability with mixed constraints.

It is worth observing that there is a one-to-one correspondence between constraints and arcs of the digraph and that the constraint $(x', y; y', x)$ associated to arc (x, y) belongs to C if and only if $\pi_0(x') > \pi_0(y')$. To prove the correspondence between solutions of AS and solutions of MBR we now exploit a basic idea: any permutation of the vertices of a digraph partitions its arcs into left-to-right and right-to-left arcs. We recall that a topological sort of an acyclic digraph yields no right-to-left arc. Based on this idea, Figure 2 points out the meaning of both crossing and non-crossing constraints: if arc (x', y') is left-to-right according to π_0, then we force x and y to have the same ordering as x' and y' using a crossing constraint (see Figure 2(a)); if arc (x', y') is right-to-left according to π_0, then we force x and y to have the opposite ordering of x' and y' using a non-crossing constraint (see Figure 2(b)).

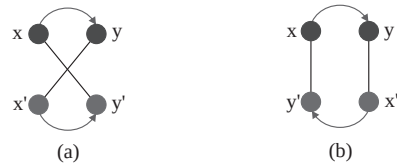

Fig. 2. Crossing and non-crossing constraints: (a) $(x', y; y', x) \in I$; (b) $(x', y; y', x) \in C$. In both cases arc (x, y) is forced to be a left-to-right arc.

Roughly speaking, our constraints force the presence of as many left-to-right arcs as possible. Proving that any permutation π_1 realizes k constraints from $I \cup C$ iff it yields an acyclic subgraph of G containing $h = k$ arcs is now easy. Let $c = (x', y; y', x)$ be a constraint from $I \cup C$. If $c \in I$ (i.e., $\pi_0(x') < \pi_0(y')$), then c is realized by π_1 if and only if $\pi_1(x) < \pi_1(y)$. Analogously, if $c \in C$ (i.e., $\pi_0(x') > \pi_0(y')$), then c is realized by π_1 if and only if $\pi_1(x) < \pi_1(y)$. In both cases the satisfaction of constraint c implies that arc (x, y) is a left-to-right arc according to π_1, and vice-versa. Therefore, any permutation π_1 satisfies a number of constraints equal to the number of left-to-right arcs that it induces.

In the example in Figure 1, choosing e.g. $\pi_1 = \langle a, b, c, d \rangle$ partitions the arcs of G as in Figure 1(c) and yields the bipartite straight-line drawing of B shown in Figure 1(d). In this drawing only constraint $(d', a; a', d) \in I$ is not satisfied; note this constraint corresponds to the unique dotted arc $(d, a) \in A \setminus A'$.

Theorem 2. *One-sided bipartite realizability is NP-hard.*

Proof. Let us consider an instance of MBR specified by a bipartite graph $B = (V_0, V_1, E)$, a set of crossing and non-crossing constraints $I \cup C$, a permutation π_0, and an integer k (w.l.o.g. in the following we assume that $\pi_0(a) < \pi_0(b)$ for any constraint $(a, x; b, y) \in I \cup C$). We build a corresponding instance of OBR specified by a bipartite graph $B' = (V_0', V_1', E')$, a set of non-crossing constraints C', a permutation π_0', and an integer k' as follows.

For each crossing constraint $(a, x; b, y) \in I$ we create a vertex named $\hat{ab}$, an edge $(\hat{ab}, y)$, and a non-crossing constraint $(\hat{ab}, y; a, x)$. We denote the sets of all the new vertices, the new edges, and the new constraints with $\hat{V_0}$, $\hat{E}$, and $\hat{C}$, respectively. We then set $V_0' = V_0 \cup \hat{V_0}$, $V_1' = V_1$, $E' = E \cup \hat{E}$, $C' = C \cup \hat{C}$, and $k' = k$. At last, we choose any permutation π_0' such that all vertices in $\hat{V_0}$ are to the left of vertices in V_0 (their relative positions are not important for our purposes), while the vertices in V_0 retain the same ordering as in π_0.

In the following we show that any permutation π_1 satisfies the same number of constraints in the instance of MBR and in the corresponding instance of OBR, proving a one-to-one correspondence between the solutions of the two problems. It is obvious that any permutation π_1 satisfies the same constraints from C in both instances. Moreover, there is a one-to-one correspondence between constraints in I and constraints in $\hat{C}$: let $(a, x; b, y) \in I$ and let $(\hat{ab}, y; a, x) \in \hat{C}$ be the corresponding non-crossing constraint. Recall that by construction we

have $\pi'_0(\hat{ab}) < \pi'_0(a) < \pi'_0(b)$. Therefore, the crossing constraint $(a, x; b, y)$ is realized by π_1 if and only if the non-crossing constraint $(\hat{ab}, y; a, x)$ is, proving that each satisfied constraint from I corresponds to a satisfied constraint from $\hat{C}$ and vice-versa.

Approximability and polynomiality. In order to prove that OBR is polynomial if $k = |C|$ and that a maximum set of realizable constraints can be approximated within a constant ratio we use an approximation preserving reduction [1] to the maximum acyclic subgraph problem. This problem is the optimization version of AS: given a digraph $H = (N, A, w)$ with positive arc weights w, it requires to find a maximum weight set of arcs $A' \subseteq A$ such that the subgraph $H' = (N, A')$ is acyclic.

Let $B = (V_0, V_1, E)$ and π_0 specify an instance of one-sided bipartite realizability. W.l.o.g. we assume that for each $(a, x; b, y) \in C$ it holds $\pi_0(a) < \pi_0(b)$; otherwise we change C by removing $(a, x; b, y)$ and adding the analogous constraint $(b, y; a, x)$. In the reduction we build a weighted *constraint digraph* $H = (N, A, w)$ as follows: $N = V_1$; $A = \{(x, y) : \exists\, a, b \in V_0 \; s.t. \; (a, x; b, y) \in C\}$; $\forall (x, y) \in A$, $w(x, y)$ is the number of constraints of C containing x and y in the second and fourth positions, respectively. In other words, the weight of arc (x, y) is the number of constraints that require $\pi_1(x) < \pi_1(y)$ in order to be realized.

An almost trivial observation is that H is acyclic iff all the constraints in C are realizable. This means that OBR can be solved in linear time if $k = |C|$, thanks to the fact that both the previous reduction and checking the acyclicity of the constraint digraph can be accomplished in linear time.

The following property also holds: each $A' \subseteq A$ such that the subgraph $H' = (N, A')$ is acyclic corresponds to a realizable set of constraints $C' \subseteq C$ with $|C'| \geq w(A')$. Indeed, if π_1 is a topological sort of H', then π_1 satisfies at least all the constraints corresponding by construction to the arcs in A', whose quantity is $w(A') = \sum_{(x,y)\in A'} w(x, y)$ (it could happen that π_1 satisfies more constraints if A' is not maximal). Finally, it is easy to see that the weight of a maximum acyclic subgraph of the constraint digraph equals the maximum number of realizable constraints, i.e., $w(A^*) = |C^*|$.

Let $r \geq 1$ be any approximation ratio for the maximum acyclic subgraph problem. From the previous considerations we have:

$$|C'| \geq w(A') \geq \tfrac{1}{r} \cdot w(A^*) = \tfrac{1}{r} \cdot |C^*|$$

proving that C' is an r-approximate solution for one-sided bipartite realizability, as well. Hence, a maximum set of realizable constraints can be approximated with ratio 2 [7].

4 Constrained Crossing Minimization

A widely used approach for creating layered drawings of directed graphs is presented in [14] and is known in literature as *hierarchical approach*. In order to produce readable drawings, different aesthetic criteria are taken into account by

this method; among them the number of edge crossings deserves special attention. In this section we show how to extend the hierarchical approach in order to support non-crossing constraints. Our extension hinges upon the approximation algorithm for one-sided bipartite realizability presented in Section 3. We first briefly recall how Sugiyama's algorithm works. Given a directed graph G, it performs four main steps. (1) *Cycle removal*: the direction of some arcs is temporarily reversed to make G acyclic. (2) *Layer assignment*: each vertex is assigned with a level so that all the arcs "flow" in the same direction. Dummy vertices are added for arcs spanning more than two levels. (3) *Crossing reduction*: vertices in each level are ordered so as to minimize the total number of edge crossings. (4) *Coordinates assignment*: vertical coordinates are proportional to layers, horizontal coordinates depend on the permutation of the vertices within each layer. At this point the direction of the reversed arcs is also restored.

In order to extend the hierarchical approach to support non-crossing constraints we need to focus on steps 2 and 3, because neither the removal of cycles nor the assignment of coordinates are affected by this kind of constraints. In the following we denote with G the directed graph to be drawn and with C the set of non-crossing constraints to be realized in the drawing.

Extending step 2. We recall that in this step the directed graph G is transformed into a proper layered graph. The set of constraints C must be therefore changed accordingly, due to the fact that dummy vertices could be added to split an edge involved in some constraint. We let the algorithm assign layers to vertices as usual, where $l(v)$ denotes the level of a vertex v after the layer assignment. Then we change C as follows.

Let (a, x) and (b, y) be any two arcs of G such that constraint $(a, x; b, y) \in C$; w.l.o.g. we can assume $l(a) < l(x)$ and $l(b) < l(y)$. If $l(x) = l(a) + 1$ and $l(y) = l(b)+1$ we do not touch C. Otherwise, at least a dummy vertex has been added to split (a, x) or (b, y) or both and constraint $(a, x; b, y)$ must be suitably replaced. Let $\langle u_k, u_{k+1}, \ldots, u_{k+s}\rangle$ be the path that replaced arc (a, x), with $u_k = a$, $u_{k+s} = x$, $k = l(a)$, and $k + s = l(x)$. Analogously, let $\langle v_h, v_{h+1}, ..., v_{h+t}\rangle$ be the path that replaced arc (b, y), with $v_h = b$, $v_{h+t} = y$, $h = l(b)$, and $h+t = l(y)$. Let also $[r, r+q] = [k, k+s] \cap [h, h+t]$. If $[r, r+q] \neq \emptyset$ we replace constraint $(a, x; b, y)$ with the constraints $(u_r, u_{r+1}; v_r, v_{r+1}) \ldots (u_{r+q-1}, u_{r+q}; v_{r+q-1}, v_{r+q})$. Otherwise we remove it, since it is going to be satisfied by any drawing obtained from layer assignment l. Figure 3 shows an example of the two cases. It is easy to get convinced that an original constraint will be realized in the final drawing if and only if all the new constraints are satisfied.

Thanks to the preprocessing of set C described above, we can now assume to deal only with constraints involving edges that join vertices in consecutive layers. More formally, we assume that G has k layers, named $L_0 \ldots L_k$, and that each $(a, x; b, y) \in C$ is such that $\exists i \in [0..k-1]$ such that $a, b \in L_i$ and $x, y \in L_{i+1}$. Based on this assumption, we can partition C into $k-1$ sets, named $C_{0,1} \ldots C_{k-1,k}$, where $C_{i,i+1}$ contains the constraints related to edges with endpoints in L_i and L_{i+1}.

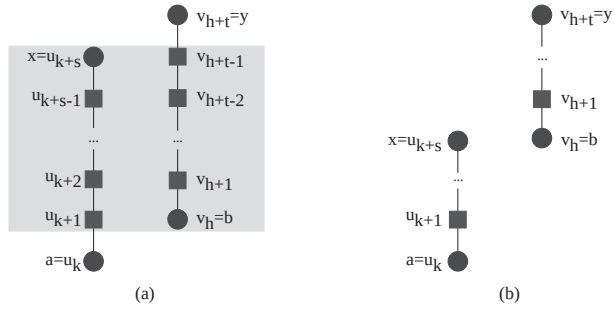

Fig. 3. Constraints replacement during the layer assignment step: dummy vertices are squared. (a) $(a, x; b, y)$ is replaced by constraints between edges in the grey band; (b) the grey band is empty: $(a, x; b, y)$ is removed from C.

Extending step 3. Before attempting at minimizing edge crossings, we introduce a *constraint realization* step, aimed at finding a large set of realizable constraints. Then we devise a crossing minimization strategy able to take into account and satisfy the set of realizable constraints previously identified.

We recall that the most used strategy for crossing minimization in hierarchical drawings is based on a layer-by-layer sweep heuristic [3], that requires to repeatedly solve a crossing minimization problem on a bipartite graph with one side fixed. We therefore adapt the layer-by-layer sweep method to support non-crossing constraints, by repeatedly solving a one-sided bipartite realizability problem as follows.

A vertex ordering π_0 for L_0 is first randomly chosen. Then, for $i = 0$ to $k-1$, the bipartite subgraph B_i of G induced by the vertices in $L_i \cup L_{i+1}$ and the set of constraints $C_{i,i+1}$ are considered. An approximated solution to OBR on this instance is found, keeping layer L_i fixed. This leads to identify a set $C'_{i,i+1} \subseteq C_{i,i+1}$ of realizable constraints. We have then to find a vertex ordering for L_{i+1} which minimizes edge crossings while realizing the constraints of $C'_{i,i+1}$. Not all crossing minimization algorithms can be easily adapted to accomplish this task (think, e.g., of the well-known median/barycenter algorithms [4,14]); here we suggest a modification of the penalty minimization technique discussed in [2].

With this method, a penalty digraph P (with vertex set L_{i+1}) is built from B_i and π_i, a feedback arc set F is found on P, and a topological sort of the vertices in the subgraph of P obtained by deleting arcs in F is returned as ordering for L_{i+1}. We recall that a feedback arc set of a directed graph is a subset of arcs whose removal makes the digraph acyclic; the feedback arc set problem requires to find a minimum weight feedback arc set.

Crossing constraints can be incorporated in the penalty based approach by suitably assigning weights to some arcs of P and by applying the previous algorithm as is. Namely, if $(a, x; b, y)$ is a constraint in $C'_{i,i+1}$ with $a, b \in L_i$ and $x, y \in L_{i+1}$, we add arc (x, y) to P if it does not already exist, and we set

$w(x, y) = \infty$. Since $C'_{i,i+1}$ is realizable, the subgraph of P induced by the infinite weight arcs is acyclic. Any approximation algorithm for the feedback arc set problem applied on P will therefore choose none of these arcs, thus minimizing crossings while satisfying all the constraints in $C'_{i,i+1}$.

As far as the algorithm above is concerned, it may happen that a pair of edges decomposed by means of dummy vertices may cross several times. This can be avoided by assigning ∞ weight to suitably chosen arcs in the constraint digraph. We also remark that our algorithm can be extended to deal with constraints concerning the relative positions of vertices within each layer. Due to the lack of space, we defer the details to the full paper.

References

1. G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Approximate Solution of NP-Hard Optimization Problems*. Springer Verlag, 1999.
2. Demetrescu, C. and Finocchi, I. Break the "right" cycles and get the "best" drawing. In B.E. Moret and A.V. Goldberg, editors, *Proc. 2nd Int. Workshop on Algorithm Engineering and Experiments (ALENEX'00)*, 171–182, 2000.
3. G. Di Battista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, Upper Saddle River, NJ, 1999.
4. P. Eades and N.C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11:379–403, 1994.
5. E.R. Gansner, E. Koutsofios, S.C. North, and K.P. Vo. A technique for drawing directed graphs. *IEEE Trans. Softw. Eng.*, 19:214–230, 1993.
6. M.R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to Theory of NP-completeness*. W.H.Freeman, 1979.
7. R. Hassin and S. Rubinstein. Approximations for the maximum acyclic subgraph problem. *Information Processing Letters*, 51:133–140, 1994.
8. S. Leipert. *Level Planarity Testing and Embedding in Linear Time*. PhD-Thesis, Universität zu Köln, 1998.
9. J. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21(4):549–568, 1974.
10. J. Kratochvíl. String graphs II: Recognizing string graphs is NP-hard. *Journal Combin. Theory Ser. B*, 52:67–78, 1991.
11. J. Kratochvíl. Crossing number of abstract topological graphs. In *Proc. 6th Int. Symp. on Graph Drawing (GD'98)*, LNCS 1547, 238–245, 1998.
12. J. Kratochvíl, A. Lubiw, and J. Nesetril. Noncrossing subgraphs of topological layouts. *SIAM Journal on Discrete Mathematics*, 4:223–244, 1991.
13. J. Opatrny. Total ordering problem. *SIAM Journal on Computing*, 8(1):111–114, 1979.
14. K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981.