Visualization of Bipartite Graphs in Limited Window Size

William Evans[1], Kassian Köck[2], and Stephen Kobourov[3]

[1] Dept. of Computer Science, University of British Columbia, Vancouver, Canada
will@cs.ubc.ca
[2] Department of Computer Science, University of Passau, Passau, Germany
koeckk@fim.uni-passau.de
[3] Department of Computer Science, University of Arizona, Tucson, Arizona, USA
kobourov@cs.arizona.edu

Abstract. Bipartite graphs are commonly used to visualize objects and their features. An object may possess several features and several objects may share a common feature. The standard visualization of bipartite graphs, with objects and features on two (say horizontal) parallel lines at integer coordinates and edges drawn as line segments, can often be difficult to work with. A common task in visualization of such graphs is to consider one object and all its features. This naturally defines a drawing window, defined as the smallest interval that contains the x-coordinates of the object and all its features. We show that if both objects and features can be reordered, minimizing the average window size is NP-hard. However, if the features are fixed, then we provide an efficient polynomial time algorithm for arranging the objects, so as to minimize the average window size. Finally, we introduce a different way of visualizing the bipartite graph, by placing the nodes of the two parts on two concentric circles. For this setting we also show NP-hardness for the general case and a polynomial time algorithm when the features are fixed.

Keywords: bipartite graphs, NP-hardness, two-layer drawing, circle layout

1 Introduction

Bipartite graphs arise in many applications and are usually visualized with 2-layer drawings, where vertices are drawn as points at integer coordinates on two distinct parallel lines, and edges are straight-line segments between their endpoints. Such drawings occur as components in layered drawings of directed graphs [19] and also as final drawings, e.g., in tanglegrams for phylogenetic trees [6,11].

A common task in the exploration of such bipartite graphs $G = (P \cup C, E)$, where P is the set of *parent* (object) vertices and C is the set of *child* (feature) vertices, is to identify the neighbors (children) of a parent vertex of interest. A typical approach is to click on this parent and highlight the edges to its children,

while hiding/shading the rest of the graph. Naturally, it is desirable that the highlighted edges fit in the display. This motivates work on placing the vertices at integer coordinates so as to minimize the maximum *window*, i.e. the smallest x-interval that contains a parent and all its children, over all parents. Bekos et al. [4] show that minimizing the maximum window size can be solved efficiently when the children C are fixed and the parents P can be placed, and is NP-hard when the parents are fixed and the children can be placed. Note that this asymmetry is due to the windows being defined only for parents P. As a side effect of the underlying greedy approach, the algorithm of Bekos et al. [4] often results in a much larger than optimal average window size in order to minimize the max window. So if the max window exceeds the display size, many parents may have windows that exceed that size. In this paper, we consider the problem of minimizing the average window size directly.

Unlike our approach, methods for constructing 2-layer drawings often try to minimize the number of edge crossings, which is an NP-hard problem even when the order of one layer is fixed [10]. Vertex splitting provides another alternative approach to reduce the number of crossings, by replacing some vertices on one layer by multiple copies and distributing incident edges among these copies [9]. In bipartite graphs arising in domain applications, such as visualizing relationships between anatomical structures and cell types in the human body [1], vertex splitting makes sense only on one side of the layout. Several variants of optimizing such layouts have been recently studied [2,17,3].

Formally, the input consists of a bipartite graph $G = (P \cup C, E)$. The output is a **2-layer drawing** of G in which the vertices in P and in C are located at distinct integer coordinates on two parallel lines ℓ_P and ℓ_C, respectively (w.l.o.g., $\ell_P : y = 1$ is the *parent layer* and $\ell_C : y = 0$ is the *child layer*). The drawing is specified by a function $x : P \cup C \to \mathbb{Z}_{\geq 0}$ which defines the x-coordinate of each vertex in the drawing. No two parents can have the same x-coordinate and neither can two children; so x is injective when restricted to P or C. The objective is to minimize the average window size of the parents in the drawing (defined by) x. The *window* $w(p)$ of a parent $p \in P$ is the smallest x-interval that contains the locations of p and its neighbors (children) in G. Its size is its length, i.e., $\max_{a,b \in S} |x(a) - x(b)|$ where $S = \{c \mid (p, c) \in G\} \cup \{p\}$. Note that the smallest window size is 0 for a parent with no children or one child that shares its parent's x-coordinate. The *span* $s(p)$ of a parent $p \in P$ is the smallest x-interval that contains the children of p ($s(p) \subseteq w(p)$). Motivated by common assumptions in layered graph drawing [8,15] we consider two variants: one when we can choose the x-coordinates of the vertices of both P and C and the other when the x-coordinates of one of them is fixed. The Minimum Average Window Problem takes as input a graph $G = (P \cup C, E)$ and a value σ and determines if G has a drawing in which the average window size of the parents is at most σ. We focus on the equivalent Minimum Window Sum Problem (MWS) which determines if the *sum* of the window sizes can be at most some given Σ.

We also consider the same problem when the drawing maps parents and children to two concentric circles: parents to the inner circle and children to the

outer circle. A **2-ring drawing** of G is specified by an integer size $r \geq 0$ and a function $x : P \cup C \to \mathbb{Z}_r$ (the integers mod r) which determines the locations of the vertices: The polar coordinates for $p \in P$ are $(1, \frac{2\pi}{r}x(p))$ and for $c \in C$ are $(2, \frac{2\pi}{r}x(c))$. Again, we require x to be injective when restricted to P or C (so $|P|, |C| \leq r$). The distance between two vertices a and b with locations specified by x is $d(a, b) = \min\{|x(a) - x(b)| \bmod r, |x(b) - x(a)| \bmod r\}$, i.e., the smallest of the clockwise or counter-clockwise distances from $x(a)$ to $x(b)$. The *window* $w(p)$ of parent $p \in P$ is the smallest interval of the circle that contains the angle locations of p and its children. Its size is measured in units of $\frac{2\pi}{r}$ radians. The *span* $s(p)$ of parent $p \in P$ is the smallest circle interval that contains the angle locations of the children of p ($s(p) \subseteq w(p)$). See Fig. 1 for an example of 2-layer and 2-ring drawing (with size $r = 7$) of the same bipartite graph with parents $A, B, \ldots, H$ and children $a, b, \ldots, h$.

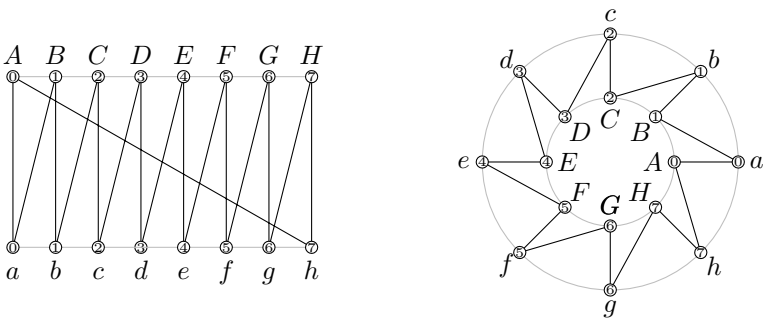


Fig. 1: An even cycle represented using a 2-layer drawing with minimum total window sum 14 and using a 2-ring drawing with minimum total window sum 8.

Our Contributions

In this paper we present the following results.

In the 2-layer setting:

- Minimizing the average window size is NP-hard when we can choose the locations of both parents and children.
- When the children are fixed, determining whether every parent can be placed in its span can be done in polynomial time.
- When the children are fixed, placing the parents to minimize the average window size can be done in polynomial time.

In the 2-ring setting:

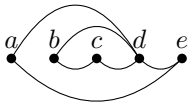
- Minimizing the average window size is NP-hard when we can choose the locations of both parents and children.
- When the children are fixed, placing the parents to minimize the average window size can be done in polynomial time.

2 The Two Layer Setting

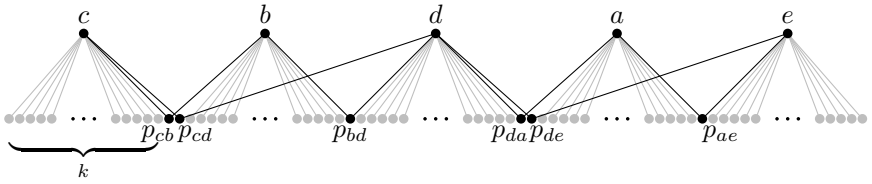
We begin by showing the NP-hardness of minimizing the average window size when both parents and children can be placed, and then provide two polynomial time algorithms for the case when the children are fixed.

2.1 Hardness of minimizing average window size.

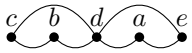
The Linear Arrangement Problem (LAP) takes as input an undirected graph $G = (V, E)$ and an integer W and decides whether there is a bijection $x : V \to \{0, 1, \ldots, |V| - 1\}$ such that $\sum_{(u,v)\in E} |x(u) - x(v)| \leq W$. In other words, LAP decides whether there is a straight-line drawing of G with vertices at integer coordinates on the x-axis so that the sum of the edge lengths is at most W. LAP is a classic NP-complete problem [13].



(a) A sample input graph to LAP.



(b) The corresponding bipartite graph instance of MWS, drawn to minimize the window sum.



(c) The optimal linear arrangement of the original graph.

Fig. 2: Example illustrating the reduction from the linear arrangement problem to the problem of minimizing the window sum.

Theorem 1. *Deciding whether a given a bipartite graph $G = (P \cup C, E)$ has a 2-layer drawing with average window size σ when vertices in both P and C can be placed is NP-complete.*

Proof. We show the equivalent statement that Minimum Window Sum (MWS) is NP-complete. MWS is in NP since it takes time polynomial in the size of the

input graph to verify that, for a given drawing $x : P \cup C \to \mathbb{Z}_{\geq 0}$, the sum of the window sizes of $p \in P$ is at most W.

To show that MWS is NP-hard, we reduce LAP to it. We construct an input $G' = (P \cup C, E')$, Σ to MWS from an input $G = (V, E)$, W to LAP as follows:

- Let k be an odd number with $k \geq 3|E|^2 + |E| + 2|V|$.
- The children C in G' are the vertices V in G.
- The parents P in G' are one *edge parent* p_{uv} (which is shorthand for $p_{\{u,v\}}$) for each $(u, v) \in E$ with edges (p_{uv}, u) and (p_{uv}, v) in G'; and k *block parents* $v_1, v_2, \ldots, v_k$ for each vertex $v \in V$, with edges $(v_1, v), (v_2, v), \ldots, (v_k, v)$ in G'.
- Let $\Sigma = |V|\frac{k^2-1}{4} + |E|^2 + kW$.

This construction takes time polynomial in the size of G. It remains to show that $G = (V, E)$ has a linear arrangement of total edge length at most W if and only if $G' = (P \cup C, E')$ has a drawing with window size sum at most Σ.

We first show that if G has a linear arrangement $v(0), v(1), \ldots, v(|V|)$ (where $v(i)$ is the ith vertex in the arrangement) of total edge length at most W then G' has a drawing with window size sum at most Σ.

We construct a bipartite drawing of G' by placing the k block parents of $v(i)$, starting with $i = 0$, consecutively (on line ℓ_P) followed (in any order) by the edge parents $p_{v(i)v(j)}$ for all edges $(v(i), v(j)) \in E$ with $j > i$, and then repeating this process for $i = i + 1$ until all block parents and edge parents are placed. We place each child $v(i)$ (on line ℓ_C) below the middle block parent of $v(i)$. Thus, the sum of the window sizes of the block parents of any child $v(i)$ is $\frac{k-1}{2} + \left(\frac{k-1}{2} - 1\right) + \cdots + 1 + 0 + 1 + \cdots + \left(\frac{k-1}{2} - 1\right) + \frac{k-1}{2} = \frac{k^2-1}{4}$ since k is odd. See Fig. 2.

The window size for an edge parent $p_{v(i)v(j)}$ (with $i < j$) is at most $(j - i)k$ for the block parents in the window plus at most $|E|$ for the edge parents in the window. The total window sum is at most:

$$|V|\frac{k^2-1}{4} + \sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} ((j-i)k + |E|) \leq |V|\frac{k^2-1}{4} + |E|^2 + kW = \Sigma$$

since the sum of $(j - i)$ over all edges is the total edge length of the linear arrangement $v(0), v(1), \ldots, v(|V|)$, which we assumed to be at most W.

We next show that if G' has a 2-layer drawing with total window sum at most Σ then G has a linear arrangement with total edge length at most W. This requires establishing several properties of any optimal drawing x of G':

Property 1. We may assume, by relabeling block parents if necessary, that if $x(u) < x(v)$ for $u, v \in C$ then $x(u_i) < x(v_j)$ for all block parents for u and v where $i, j \in [k]$.

Proof. Suppose $x(u_i) > x(v_j)$ for some $i, j \in [k]$. If we switch u_i and v_j, the sum of the window sizes of these two block parents remains the same if u_i and v_j are both to the left of u or both to the right of v, and decreases otherwise.

Property 2. Child v lies strictly within the x-interval of its block parents, i.e., for all $v \in C$, $x(v_1) < x(v) < x(v_k)$, where we have assumed (by renumbering) that v_i is the ith leftmost block parent of v in the realization.

Proof. Suppose $x(v) \leq x(v_1)$ (a symmetric argument applies when $x(v_k) \leq x(v)$). There must be an empty spot s for v with $s \in \{x(v_1)+1, x(v_1)+2, \ldots, x(v_1)+|V|\}$ since there are only $|V|$ children in G' and $x(v)$ is not in that set. Moving v from $x(v)$ to s can increase the window size of any edge parent by at most $s - x(v)$. It can increase the window size of any block parent p with $x(v_1) \leq x(p) < s$ by at most $s - x(v)$ as well. It *decreases* the window size of any block parent p with $s \leq x(p) \leq x(v_k)$ by at least $s - x(v)$. Since $k > |E| + 2|V|$, there are more than $|E| + |V|$ block parents whose windows decrease (by at least $s - x(v)$) and at most $|E|$ edge parents plus at most $|V|$ block parents whose windows increase (by at most $s - x(v)$) as a result of moving child v to s. Thus the sum of the window sizes decreases and the original realization cannot be optimal.

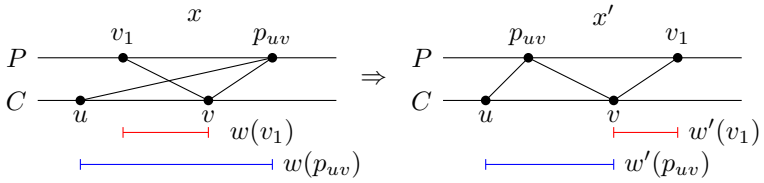


Fig. 3: Switching edge parent p_{uv} with leftmost block parent v_1 of v so that p_{uv} lies between u and v.

Property 3. The location $x(p_{uv})$ of edge parent p_{uv} lies strictly between the locations of u and v.

Proof. We may assume by renaming that $x(u) < x(v)$. Suppose $x(v) \leq x(p_{uv})$ (a similar symmetric argument applies when $x(p_{uv}) \leq x(u)$). Switch the locations of the leftmost block parent v_1 of v with p_{uv} to obtain a new drawing x' with window sizes w' (see Figure 3). Note that x' differs from x only in that $x'(v_1) = x(p_{uv})$ and $x'(p_{uv}) = x(v_1)$. Since (by Properties 1 and 2) v_1 lies strictly between u and v in realization x, the new location of p_{uv} in realization x' lies strictly between u and v, decreasing the window size of p_{uv} by $x(p_{uv}) - x(v)$. The new window size of v_1 is at most $x(p_{uv}) - x(v)$ and since the original window size of v_1 is at least 1 (by Property 2), the sum of window sizes in realization x' is less than in x. Thus, x cannot be optimal.

Property 4. The parents P occupy $k|V| + |E|$ consecutive locations.

Proof. Suppose there is an empty spot s with parents to the left and right of s. No child v has $x(v) = s$ otherwise we could move a block parent of v to location

s and decrease the window size sum. Thus we can decrement the location of all parents and children located to the right of s by 1 to create a new realization (no two parents or two children at the same location) with no increase in window size sum, and a strict decrease if G is connected.

Property 5. The block parents of v are consecutive, i.e., $x(v_{1+i}) = x(v_1) + i$ for all $i \in [k-1]$ and $v \in V$.

Proof. Suppose an edge parent p_{uv} is in the x-interval of the block parents of a vertex t, i.e., $x(t_1) < x(p_{uv}) < x(t_k)$. It can't be in the x-interval of the block parents of more than one vertex by Property 1. By Property 3, $x(u) \leq x(p_{uv}) \leq x(v)$. Swapping p_{uv} with t_1 or t_k thus keeps p_{uv} in its span (since at least one of t_1 or t_k is in the span of p_{uv}), but out of the x-interval of any vertex's block parents. It also decreases the window size of the swapped block parent (t_1 or t_k) by at least one.

With these properties established, we continue with the proof of the theorem. Let x be an optimal 2-layer drawing of G' with window sum at most Σ. We claim that the order $v(0), v(1), \ldots, v(|V|-1)$ of the children on line ℓ_C (where $x(v(i)) < x(v(j))$ for all $i < j$) is a linear arrangement of the vertices of G with total edge length at most W.

By Property 4, we may assume that the parents occupy $k|V|+|E|$ consecutive locations and, by Property 5, that the block parents of each child v form a consecutive x-interval in this sequence. Furthermore, by Property 2, child v lies within the x-interval of its block parents. Thus the window of any edge parent $p_{v(i)v(j)}$ $(i < j)$ extends from some position in the x-interval of the block parents of $v(i)$ to some position in the x-interval of the block parents of $v(j)$. To minimize the sum of the window sizes of the block parents of child v requires placing v in the middle of the x-interval of its block parents, resulting in a total contribution of v's block parents to the window sum of at least $\frac{k^2-1}{4}$. To minimize the sum of all window sizes, including the windows of edge parents, an optimal drawing may place v at a location that is not in the middle of the x-interval of its block parents. However, if v is placed at distance d from the middle, the sum of the window sizes of the block parents of v increases by d^2 (by symmetry, the edge lengths after v is moved are the same as the lengths before the move, except for the lengths to the furthest d block parents after the move, which each increase by d), while the decrease in the window sum of all $|E|$ edge parents is at most $d|E|$. Thus, v must be placed at distance $d \leq |E|$ from the middle of its block parents in an optimal drawing. Even with this placement of v, the window size of $p_{v(i)v(j)}$ is at least

$$(j-i-1)k + 2\left(\frac{k-1}{2} - d\right) \geq (j-i)k - (2|E|+1).$$

The total contribution of block parents and edge parents to the window sum is thus at least

$$|V|\frac{k^2-1}{4} + \sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} ((j-i)k - 2|E| - 1)\,.$$

Since this is at most $\Sigma = |V|\frac{k^2-1}{4} + |E|^2 + kW$, we know

$$|V|\frac{k^2-1}{4} + \sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} ((j-i)k - 2|E| - 1) \leq |V|\frac{k^2-1}{4} + |E|^2 + kW$$

and thus

$$\sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} (j-i)k \leq 3|E|^2 + |E| + kW.$$

Since $k \geq 3|E|^2 + |E| + 2|V|$,

$$\sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} (j-i) \leq W + 1 - \frac{2|V|}{k}$$

and since the sum and W are integers,

$$\sum_{\substack{\{v(i),v(j)\}\in E \\ i<j}} (j-i) \leq W.$$

So G has a linear arrangement with total edge length at most W.

2.2 Minimizing average window size for fixed children.

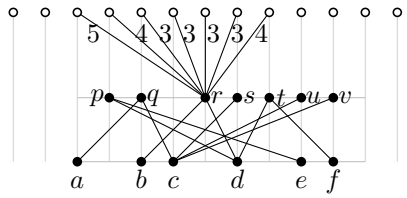
When the children are fixed, the best possible solution to the minimum average window problem is to place every parent in the span of its children. We first observe how to efficiently test if such a solution exists. After that, we show that even when it is not possible to place every parent in the span of its children we can still find a solution that minimizes the average window size in polynomial time.

Placing parents in their span. If the children C of a bipartite graph $G = (P \cup C, E)$ have already been placed at distinct integer x-coordinates then the span of every parent p is fixed: $s(p) = [\mathrm{lo}(p), \mathrm{hi}(p)]$ where $\mathrm{lo}(p)$ is the smallest x-coordinate of a child of p and $\mathrm{hi}(p)$ is the largest x-coordinate of a child of p. Our problem is to determine if it is possible to place each parent at an integer coordinate within its span without placing two parents at the same location. This is an instance of a matching problem in a *convex* bipartite graph $A = (P \cup S, F)$ where $F = \{(p, \ell) | p \in P, \ell \in s(p)\}$ and $S = \bigcup_{p\in P} s(p)$ (so S is a set of integers).

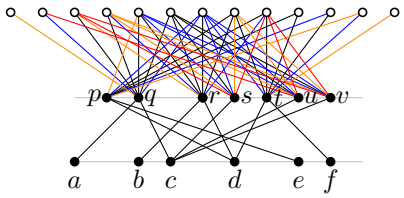
The bipartite graph $A = (P \cup S, F)$ is convex (in S) since there is an ordering of S (the integer order "$<$" in this case) such that if $(p, a) \in F$ and $(p, c) \in F$ then $(p, b) \in F$ for $b \in S$ with $a < b < c$. Graph A is defined by the parents P and the pairs $\text{lo}(p)$, $\text{hi}(p)$ for $p \in P$, which in our case can be calculated from G in $O(|E|)$ time.

The problem of finding a maximum matching in a convex (in S) bipartite graph $(P \cup S, F)$ has a long history starting with Glover's $O(|P||S|)$ time algorithm from 1967 [14] and ending with the $O(|P|)$ time algorithm of Steiner and Yeomans [18]. Using the latter algorithm, we observe that:

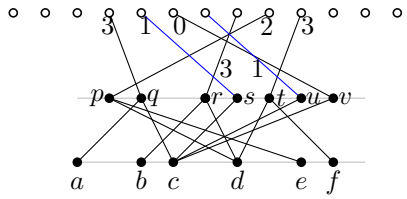
Observation 1 *Given a bipartite graph $G = (P \cup C, E)$ where the x-coordinates of the children C are fixed, finding a placement of parents $p \in P$ where every parent is in its span, or determining that no such placement exists, takes $O(|P| + |E|)$ time.*



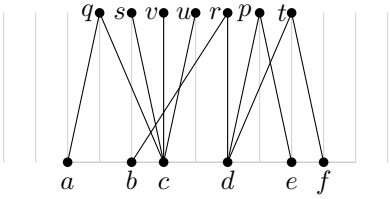
(a) Shown above the graph G are the edges matching parent vertex r to its possible locations $\circ$ in an optimal drawing of G. The weight of edge (r, ℓ) is the size of r's window if r were at location ℓ.



(b) The entire bipartite matching graph shown above the graph G. Black edges have weight equal to the parent's window size. Blue have that weight + 1. Orange: + 2. Red: + 3.



(c) A min-weight maximum size matching of parents to locations.



(d) The resulting optimal drawing with window sum 13.

Fig. 4: Using min-weight maximum size matching to find an optimal drawing when children are fixed.

Minimizing average window size. If we cannot place every parent in its span, we would like to place parents as close to their span as possible to minimize their average window size. We adopt a similar technique of finding a matching in a convex bipartite graph to determine the parents' locations. However, in this

case the edges in the matching are weighted and we ask for a minimum weight maximum matching.

Theorem 2. *Minimizing the average window size of a 2-layer drawing of a given bipartite graph $G = (P \cup C, E)$, when the locations of the children C are fixed, but the parents P can be placed, can be accomplished in $O(|P|^{2+o(1)} + |E|)$ time.*

Proof. Given the fixed locations of the children C, for each parent $p \in P$, compute the x-coordinate of the midpoint between its children's smallest and largest locations, and let $M(p)$ be the set of $|P|$ potential parent locations closest to this midpoint. We do not need to place p outside of $M(p)$ when minimizing the average window size for p since $M(p)$ contains the locations that result in the $|P|$ smallest window sizes for p and we're guaranteed to find at least one empty spot among them since there are only $|P| - 1$ other parents.

We construct a weighted bipartite graph B between parents P and their possible locations $M(P) = \bigcup_{p \in P} M(p)$ with edges (p, ℓ) for every parent p and $\ell \in M(p)$. (See Fig. 4.) There are exactly $|P|^2$ edges in the graph B and each edge (p, ℓ) has weight corresponding to the size of parent p's window if p were placed at location ℓ. The construction of B takes $O(|E| + |P|^2)$ time. We then find a min cost matching in B using the algorithm of Chen et al. [7] in time $O(|P|^{2+o(1)})$.

3 The Two Ring Setting

In this setting we are given two concentric circles, with the parents on the inner circle and the children on the outer circle.

The problems here are similar to the two layer setting, but sufficiently different that neither result in the previous section directly works here. It might appear that we can extract an optimal 2-layer drawing x of $G = (P \cup C, E)$ from an optimal 2-ring drawing x' of G by simply setting $x(v) = x'(v)$ for all $v \in P \cup C$. However, Figure 1 illustrates a difficulty: the 2-ring setting allows us to measure the distance from a parent to its child in two ways, clockwise or counter-clockwise around the inner ring. This creates the possibility of a smaller window sum in the 2-ring setting than in the 2-layer setting. We can do such an extraction if the 2-ring drawing has a point at which we can "split" the two circles and straighten them into two parallel lines.

Definition 1. *A 2-ring drawing x of a graph $G = (P \cup C, E)$ of size r is* splittable at $s \in \mathbb{Z}_r$ *if $s + 1/2$ (and thus the open interval $(s, s + 1 \bmod r)$) is not in the window $w(p)$ for any $p \in P$.*

Remark 1. A 2-ring drawing x' of $G = (P \cup C, E)$ of size r that is splittable at s can be converted into a 2-layer drawing x with the same window sum by setting $x(v) = (x'(v) - s) \bmod r$ for all $v \in P \cup C$.

Theorem 3. *Deciding whether a given bipartite graph $G = (P \cup C, E)$ has a 2-ring drawing with average window size σ when vertices in both P and C can be placed is NP-complete.*

Proof. We show the equivalent statement that the window sum version of the problem, with target sum Σ, is NP-complete. The problem is in NP since it takes polynomial time to verify that, for a given drawing $x : P \cup C \to \mathbb{Z}_r$, the sum of the window sizes of $p \in P$ is at most Σ. We reduce the 2-layer version, shown to be NP-complete in the proof of Theorem 1, to this setting by making every non-splittable 2-ring drawing of G prohibitively expensive. We do this by adding $k = |P|(|P| + |C|) + 1$ independent edges to G.

Given an input $G = (P \cup C, E)$, Σ to the 2-layer minimum sum problem, we create a new graph $G' = (P' \cup C', E')$ where $P' = P \cup \{u_1, u_2, \ldots, u_k\}$, $C' = C \cup \{v_1, v_2, \ldots, v_k\}$, and $E' = E \cup \{(u_i, v_i) | i \in [k]\}$. We claim that G has a 2-layer drawing with window sum at most Σ if and only if G' has a 2-ring drawing with window sum at most Σ.

If G has a 2-layer drawing x with window sum at most Σ then it is safe to assume by translating that the drawing places vertices at x-coordinates in $[0, 1, \ldots, |P| + |C| - 1]$. Otherwise, if the drawing x exceeds this interval then there is a common empty spot s in both the parents and children in x and decreasing the positions of parents and children to the right of s by one results in a drawing whose window size sum is at most the original sum. We construct a 2-ring drawing x' from x by setting $r = |P| + |C| + k$ and $x'(v) = x(v)$ for all $v \in P \cup C$ and $x'(u_i) = x'(v_i) = |P| + |C| + i - 1$ for all $i \in [k]$. Since the window sum for all parents u_i is zero the window sum of the 2-ring drawing is at most Σ.

If G' has a 2-ring drawing x' of size r with window sum at most Σ then if x' is splittable at s, Remark 1 implies we can extract a 2-layer drawing x of G from it with window sum at most Σ by setting $x(v) = (x'(v) - s) \bmod r$ for all $v \in P \cup C$. The addition to G of the (many) independent edges (u_i, v_i) to create G' ensures that the optimal drawing of G' when restricted to the vertices of G is splittable. Let x^* be this optimal 2-ring drawing of G. Suppose, for the sake of contradiction, that x^* is not splittable, then every interval between adjacent children in x^* is contained in the window $w^*(p)$ for some parent $p \in P$. Since the children of the independent edges lie in these intervals, the sum of the window sizes in the 2-ring drawing x^* is at least k. Since x^* is a subdrawing of x', the window sum of drawing x' of G' is also at least k. However, k is chosen to be larger than the window sum of the optimal 2-ring drawing of any graph G with $|P|$ parents and $|C|$ children along with k independent edges. To see this, note that one possible 2-ring drawing of such a graph starts with a 2-layer drawing of G in which every parent has a window of size at most $|P| + |C|$; followed by k independent edges each with window size 0. Placing these vertices, in order around two concentric circles of size $|P| + |C| + k$ creates a 2-ring drawing with window sum of $|P|(|P| + |C|) < k$. Thus the 2-ring drawing x' of G' is splittable and by Remark 1 G has a 2-layer drawing of size Σ.

Theorem 4. *Minimizing the average window size of a 2-ring drawing of size r of a given bipartite graph $G = (P \cup C, E)$, when the locations of the children C are fixed, but the parents can be placed, takes $O(|P|^{2+o(1)})$ time.*

Proof. This is identical to the proof of Theorem 2 with the understanding that ring size r is large enough to place the parents P and that the window size associated with a potential location of $p \in P$ is measured in the ring setting, i.e. as the smallest angle of a sector containing that potential location and p's children measured in units of $\frac{2\pi}{r}$ radians.

4 Conclusions and Open Problems

The visualization of bipartite graphs on displays of limited size is motivated by real-world applications. Minimizing the largest window size of any parent provides a reasonable solution but only if that largest size is at most the display size. We consider minimizing the average window size. Our solution may not obey the display size limit for all parents (even if such a solution exists), but it produces smaller parent windows on average, which might be preferable. We showed that in the general setting the problem is NP-hard, while for more restricted settings we provide efficient polynomial time algorithms.

The matching re-formulation of the problem is broad enough to be applied to several related problems in both the 2-layer and 2-ring settings. For example, in the 2-ring setting when the children are fixed, we can find an optimal solution using the same graph matching formulation as in Theorem 4 but instead of using a solution to the min cost matching problem on graph B, we use a solution technique for the bottleneck matching problem. This, in effect, replaces the min-average computation with a min-max computation.

Observation 2 *In the 2-ring setting, minimizing the maximum window size over all parents for a given bipartite graph $G = (P \cup C, E)$, when the size of the ring r and the children C are fixed, but parents P can be placed, takes $O(|P| \log |P| + |E|)$ time.*

Proof. Let B be the graph defined in the proof of Theorem 2 for the ring size r. If we restrict this graph to edges of weight at most w, we obtain a *circular convex bipartite graph* $B(w)$ for which the neighbors of a parent $p \in P$ form a contiguous interval around the ring. For such graphs, Liang and Blum [16] describe an algorithm to find a maximum matching based on two runs of a maximum matching algorithm for a regular convex bipartite subgraph of this graph. Using a binary search technique credited to Bhat [5] by Gabow and Tarjan [12], we can find the smallest $w \in [0, |P|]$ so that $B(w)$ contains a maximum matching (of size $|P|$ in our case). After a single $O(|E|)$-time preprocessing step, the running time is $O(|P|)$ for each test of whether the restricted graph $B(w)$ contains a maximum matching [16,18]. The total time is thus $O(|P| \log |P| + |E|)$.

We conclude with two natural open problems. First, is there a faster algorithm for minimizing the average window size when children are fixed but parents can be placed? Second, what is the complexity of minimizing the average window size when parents are fixed, but children can be placed?

References

1. URL: `https://hubmapconsortium.github.io/ccf-asct-reporter/`.
2. Reyan Ahmed, Patrizio Angelini, Michael A Bekos, Giuseppe Di Battista, Michael Kaufmann, Philipp Kindermann, Stephen Kobourov, Martin Nöllenburg, Antonios Symvonis, Anaïs Villedieu, et al. Splitting vertices in 2-layer graph drawings. *IEEE Computer Graphics and Applications*, 2023.
3. Jakob Baumann, Matthias Pfretzschner, and Ignaz Rutter. Parameterized complexity of vertex splitting to pathwidth at most 1. *arXiv preprint arXiv:2302.14725*, 2023.
4. Michael A Bekos, Henry Förster, Michael Kaufmann, Stephen Kobourov, Myroslav Kryven, Axel Kuckuk, and Lena Schlipf. On the 2-layer window width minimization problem. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 209–221. Springer, 2023.
5. K. V. S. Bhat. An $O(n^{2.5} \log_2 n)$ time algorithm for the bottleneck assignment problem. unpublished. AT&T Bell Laboratories, Napiendle, IL, 1984.
6. Kevin Buchin, Maike Buchin, Jaroslaw Byrka, Martin Nöllenburg, Yoshio Okamoto, Rodrigo Silveira, and Alexander Wolff. Drawing (complete) binary tanglegrams. *Algorithmica*, 62(1–2):309–332, 2012. `doi:10.1007/s00453-010-9456-3`.
7. Li Chen, Rasmus Kyng, Yang P Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 612–623. IEEE, 2022.
8. Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.
9. Peter Eades and CFX de Mendonça N. Vertex splitting and tension-free layout. In *Graph Drawing: Symposium on Graph Drawing, GD'95 Passau, Germany, September 20–22, 1995 Proceedings 3*, pages 202–211. Springer, 1996.
10. Peter Eades and Nicholas C Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994.
11. Henning Fernau, Michael Kaufmann, and Mathias Poths. Comparing trees via crossing minimization. *J. Comput. Syst. Sci.*, 76(7):593–608, 2010. `doi:10.1016/j.jcss.2009.10.014`.
12. H.N. Gabow and R.E. Tarjan. Algorithms for two bottleneck optimization problems. *J. Algorithms*, pages 411–417, 1988.
13. M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1(3):237–267, 1976.
14. F. Glover. Maximum matching in convex bipartite graphs. *Naval Research Logistic Quarterly*, pages 313–316, 1967.
15. Michael Kaufmann and Dorothea Wagner, editors. *Drawing Graphs, Methods and Models*, volume 2025 of *LNCS*. Springer, 2001. `doi:10.1007/3-540-44969-8`.
16. Y. D. Liang and N. Blum. Circular convex bipartite graphs: Maximum matching and Hamiltonian circuits. *Information Processing Letters*, 56(4):215–219, 1995.
17. Martin Nöllenburg, Manuel Sorge, Soeren Terziadis, Anaïs Villedieu, Hsiang-Yun Wu, and Jules Wulms. Planarizing graphs and their drawings by vertex splitting. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Graph Drawing and Network Visualization (GD'22)*, volume 13764 of *LNCS*. Springer, 2022.
18. G. Steiner and J.S. Yeomans. A linear time algorithm for determining maximum matchings in convex, bipartite graphs. *Computers and Mathematics with Applications*, pages 91–96, 1996.

19. Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybern.*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.