A Polynomial time Algorithm for the Maximum Weight Independent Set Problem on Outerstring Graphs[*]

J. Mark Keil[†], Dinabandhu Pradhan and Martin Vatshelle
keil@cs.usask.ca, dina@cs.usask.ca, vatshelle@ii.uib.no
Department of Computer Science
University of Saskatchewan, Canada

Abstract

Outerstring graphs are the intersection graphs of curves that lie inside a disk such that each curve intersects the boundary of the disk in one of its endpoints. Outerstring graphs were introduced in 1991 and are amongst the most general classes of intersection graphs studied, including among others, chordal graphs and interval filament graphs. To date no polynomial time algorithm is known for any of the classical graph optimization problems on outerstring graphs, in fact most are NP-hard. It is known that there is an intersection model for any outerstring graph that consists of polygonal arcs attached to a circle. However, this representation may require an exponential number of segments relative to the size of the graph.

We develop a general simple dynamic programming algorithm for the Maximum Weight Independent Set problem. Given an outerstring graph and an intersection model consisting of polygonal arcs with a total of N segments, we show that our algorithm solves the Maximum Weight Independent Set problem in $O(N^4)$ time. If the polygonal arcs are restricted to single segments, then outersegment graphs result. In this special case we show that our algorithm solves the Maximum Weight Independent Set problem in $O(n^3)$ time where n is the number of vertices in the graph.

[*]Supported by NSERC
[†]Corresponding Author

1 Introduction

Outerstring graphs introduced in 1991, are amongst the most general classes of intersection graphs studied. To date no polynomial time algorithm is known for any of the classical graph optimization problems on outerstring graphs even if the representation is part of the input, in fact most of the problems are NP-hard. When given an intersecion model of size N, we solve the Maximum Weight Independent Set problem for outerstring graphs in $O(N^4)$ time.

A graph G is a *geometric intersection graph* if the vertex set of G is a set of geometric objects and two such objects are adjacent in G if and only if they intersect. An *independent set* in a geometric intersection graph corresponds to disjoint geometric objects in the intersection model. The Maximum (Weight) Independent Set problem in intersection graphs of geometric objects in the plane has many applications such as train dispatching [3], map labelling [18], and data mining [8].

String graphs are the intersection graphs of curves in the plane and they are among the most general geometric intersection graphs that have been studied. String graphs are a superclass of planar graphs [21], chordal graphs, cocomparabillity graphs [16], subtree filament graphs [9] and circle graphs. Indeed the intersection graph of any collection of connected sets in the plane is a string graph. As early as 1959, Benzer [1] encountered string graphs in his study of genetic structures. Since then they have been extensively studied and have many applications. Kratochvíl et al. [12] showed that every string graph can be realized by a familty of polygonal arcs with a finite number of intersections. However in 1991, Kratochvíl and Matousek [14] constructed string graphs on n vertices that require at least 2^{cn} intersection points in any realization. This also implies that a representation of a string graph with a family of polygonal arcs may require an exponential number of bends in the polygonal arcs. In 1991, Kratochvíl [11] proved that the problem of recognizing string graphs is NP-hard, but more than two decades passed before Schaefer et al. [20] showed that recognizing string graphs was in NP.

It was in 1966 that Sinden [21] showed that all planar graphs are string graphs, thus the Maximum (Weight) Independent Set problem became known to be NP-hard on string graphs when it was proven to be NP-hard in planar graphs. Recently, Fox and Pach [5] provided approximation algorithms and exact subexponential algorithms for the Maximum Independent Set problem in string graphs. In 1976, the 3-Colorability problem for string graphs was proven NP-complete by Ehrlich et al. [19], even when a geometric representation is given as the input. The Maximum Clique problem has long been known to be NP-hard [15, 17] on string graphs. Indeed most of the classical NP-hard graph optimization problems remain NP-hard when restricted to string graphs, even when given a geometric representation.

It seems that one must somehow restrict string graphs to achieve polynomial time algorithms. The two most natural ways to restrict string graph are to either restrict the shapes of the strings, or to restrict the positions of the strings. The most commonly studied such restrictions are to restrict the strings to be straight line segments or to require that each string touches the infinite face of the plane. We first consider each of these restriction separately and then the case combining them.

Segment graphs are the intersection graphs of line segments in the plane. This restriction to line segments still allows the graphs to be useful in many applications, but unfortunately most of the classical NP-complete graph problems remain intractable on segment graphs. The NP-completeness result of Even et al. [19] on 3-Colorability also applies to segment graphs. Katochvíl and Nesetril [15] proved that the Maximum Independent Set problem in segment graphs is NP-hard even if all the segments are restricted to lie in at most two directions in the plane. It has recently been shown that the Maximum Clique problem is NP-hard on segment graphs [2]. Thus even a severe limiting of the shapes of the strings in a string graph does not lead to polynomial time algorithms.

The restriction that each string touches the infinite face of the plane was explored in 1991 [10] by Kratochvíl who defined outerstring graphs to be the intersection graphs of curves that lie inside a disk such that each curve intersects the boundary of the disc in one of its endpoints. Although outerstring graphs have been studied for more than 20 years [6, 7, 10, 13], when we consider the classical NP-hard graph optimization problems on outerstring graphs, we again do not find known polynomial time algorithms. For outerstring graphs the NP-completeness of Minimum Clique Cover, Colorability, Minimum Dominating Set, and Hamiltonian Cycle follow from the fact that they contain circle graphs. The Maximum Clique problem was recently shown to be NP-hard on ray graphs [2] which is also a subclass of outerstring graphs. To date, the Maximum Independent Set problem remains open. Recently, in a paper related to train dispatching in railways, the problem was again raised as open by Flier et al. [3].

Including both natural restrictions to strings graphs, Flier et al. [4], motivated by train dispatching, study the intersection graphs of segments lying inside a disk having one endpoint attached to the boundary of the disk, called *outersegment graphs*. Applying the additional restriction that each segment is either horizontally or vertically aligned, they are able to obtain a polynomial time algorithm for the Maximum Independent Set problem when a geometric representation of the graph is given as part of input. The Maximum Independent Set problem on general outersegment graphs is still open. The Maximum Clique problem remains NP-hard on outersegment graphs, as they also include ray intersection graphs [2]. Likewise, since outersegment graphs contain circle graphs, Minimum Clique Cover, Colorability, Minimum Dominating Set, and Hamiltonian Cycle remain NP-hard.

In this paper, we distill the essense of a simple dynamic programming approach to the Maximum Weight Independent Set problem in a graph. The approach is encapulated in an algorithm that can be used for any graph to find an independent set of vertices, and if certain conditions of input are met, the algorithm will solve the Maximum Weight Independent Set problem. In Section 2.1, we show that our algorithm can be used to solve the Maximum Weight Independent Set problem on outersegment graphs in $O(n^3)$ time if the segment representation is provided as part of the input. In Section 2.2, we show that our algorithm can solve the Maximum Weight Independent Set problem in outerstring graphs in polynomial time in the size of the input, where the input is the graph and the outerstring intersection model given as a family of polygonal arcs.

2 Algorithms

We present a simple general polynomial time algorithm, based on a dynamic programming approach, which always find an independent set and if certain conditions are met, can be used to solve the Maximum Weight Independent Set problem optimally. The idea is to solve a subproblem based on the solutions to two smaller subproblems which do not conflict. The results in this paper heavily depend on the definition of a "Conflict-free function". By suitably normalizing the input we will eventually be able to use this simple algorithm to solve the Maximum Weight Independent Set problem optimally on outerstring graphs.

In order to make the algorithm as general as possible, we first give a technical definition that we use in the main definition of a Conflict-free function which is used in Algorithm 1.

Definition 2.1 (Reach extension). *For a finite set U and any function $F : U \times U \to 2^U$, denote the reach extension of F as $RF : U \times U \to 2^U$. Define $RF(u, v)$ as the minimal set S such that:*

$$S = F(u, v) \cup \bigcup_{x \in F(u,v)} (RF(u, x) \cup RF(x, v))$$

To prove that the reach extension is uniquely defined we see that there is an equivalent definition via reachability in a graph. Define a directed graph H with one node for each pair $u, v \in U$ and for each $x \in F(u,v)$ add the arcs $uv \to ux$ and $uv \to xv$. The *reachability* in H, denoted by $reach(uv)$, is all $u'v'$ reachable from uv by a directed path in H. It is easy to see that $R(u,v) = \bigcup_{u'v' \in reach(uv)} F(u',v')$. Note that a consequence of this definition is that $\forall x \in F(u,v)$, we have $RF(u,x) \cup RF(x,v) \subseteq RF(u,v)$. We now present the main definition of this paper.

Definition 2.2 (Conflict-free function)**.** *Let G be a graph and $C : V(G) \times V(G) \to 2^{V(G)}$ be a function. Let RC be the reach extension of C, see Definition 2.1. We say that C is conflict-free if for each pair u, v of nonadjacent vertices, it satisfies the following two conditions:*

- *$RC(u,v) \cap N[u,v] = \emptyset$.*
- *$\forall x \in C(u,v)$, $N[RC(u,x)] \cap RC(x,v) = \emptyset$.*

Note that if C is a conflict free function and RC the reach extension of C then for all $u, v \in V(G)$ and all $x \in C(u,v)$, we have $|RC(u,x) \cup RC(x,v)| < |RC(u,v)|$. We now describe a simple algorithm which can be used to solve the Maximum Weight Independent Set problem. The algorithm takes as input a graph G with vertex weights ($w(v)$ for each $v \in V(G)$) and a conflict-free function C. We will fill a table M using dynamic programming. An entry $M[u,v]$, where $(u,v) \notin E(G)$ will hold the weight of an independent set (sum of weights of vertices present in the independent set) in $G[RC(u,v) \cup \{u,v\}]$, however it will not always be an independent set of maximum weight.

Algorithm 1 Weighted Independent Set by a conflict-free function

```
Input: A graph $G$ and a conflict-free function $C : V(G) \times V(G) \to 2^{V(G)}$
Output: The weight of an independent set in $G$;

$\forall (u,v)$, initialize $M[u,v] = -\infty$;
for $i = 0$ to $|V(G)|$ do
  for $u, v \in V(G)$ such that $u \neq v$ and $(u,v) \notin E(G)$ and $|RC(u,v)| = i$ do
    Set $M[u,v] = w(u) + w(v)$;
    for $x \in C(u,v)$ do
      Set $M[u,v] = max(M[u,v], M[u,x] + M[x,v] - w(x))$;
    end for
  end for
end for
return max entry in $M$.
```

Theorem 2.3. *Let G be a graph, C be a conflict-free function, and RC be the reach extension of C. Algorithm 1 returns the weight of an independent set in G in $O(n^3 + T)$ time, where T is the total time for computing $C(u,v)$ and $RC(u,v)$ for each pair of nonadjacent vertices.*

Proof. We will prove that the table entry $M[u,v]$, for all pairs $u, v \in V(G)$ where $(u,v) \notin E(G)$, stores the weight of an independent set in the graph $G[RC(u,v) \cup \{u,v\}]$. We do the proof by induction on $|RC(u,v)|$. The base cases are all pairs u, v where $RC(u,v) = C(u,v) = \emptyset$ and these are set to $w(u) + w(v)$ in the first iteration of the "for" loop and never updated. Since by definition $(u,v) \notin E(G)$, all base cases are correct.

Let $u, v \in V(G)$ be two nonadjacent vertices and let $i = |RC(u,v)|$. Assume that the table is correct for all pairs u', v' with $|RC(u',v')| < i$ and that $i > 0$. The value in $M[u,v]$ must have been created by some $x \in C(u,v)$, and $\{u,x,v\}$ must be an independent set. By Definition 2.1 and 2.2, we know that $|RC(u,x) \cup RC(x,v)| < |RC(u,v)|$ and hence by the inductive assumption, $M[u,x]$ and $M[x,v]$ are the weights of appropriate independent sets. Let I_u and I_v be the independent sets whose weights are stored in $M[u,x]$ and $M[x,v]$ respectively. We need to show that $I = I_u \cup I_v$ is an independent set in $G[RC(u,v) \cup \{u,v\}]$.

By assumption $I_u \subseteq RC(u,x) \cup \{u,x\}$ and $I_v \subseteq RC(x,v) \cup \{x,v\}$ and $x \in C(u,v)$. As a consequence of Definition 2.1, we know that $RC(u,x) \cup RC(x,v) \subseteq RC(u,v)$ and hence $I \subseteq RC(u,v) \cup \{u,v\}$. From the first condition of Definition 2.2, we get that $N[u,x,v] \cap (I \setminus \{u,x,v\}) = \emptyset$. From the second condition of Definition 2.2, we get that $N[I_u \setminus \{x\}] \cap I_v = \emptyset$, hence I is an independent set of $G[RC(u,v) \cup \{u,v\}]$. By induction every entry in M is the weight of some independent set in G.

We can precompute all values for the function C in T time such that lookup can be done in constant time, computing $RC(u,v)$ can be done in $O(n^3)$ time by building the directed acyclic graph representing the recursive calls made according to Definition 2.1. There are $O(n^2)$ entries in the table which are filled, initializations take constant time and updates take $O(n)$ time once we are given the set $C(u,v)$, hence the total runtime is $O(n^3 + T)$. □

We will now show an example of a conflict-free function, this definition will not be important for the results of this paper, but will help provide intuition for conflict free functions.

Definition 2.4 ($S_\sigma(u,v)$)**.** *Let G be a graph and σ an ordering of $V(G)$. For any pair of nodes $u, v \in V(G)$, we define the following set:*

$$S_\sigma(u,v) = \{x : \forall y \in (N(x) \setminus N(u,v) \cup \{x\}), \sigma(u) < \sigma(y) < \sigma(v)\}$$

Theorem 2.5. *For all graphs G and all orderings σ of $V(G)$, the function $S_\sigma(u,v)$ is conflict-free.*

The proof of this theorem is moved to the appendix.

2.1 Outersegment graphs

Definition 2.6 (Outersegment graph)**.** *An outersegment graph is the intersection graph of a set of line segments R such that each segment has one endpoint on a circle B and the rest of the segment is completely in the interior of B. Such a set R of segments is an outersegment representation of G.*

Let G be an outersegment graph and R be an outersegment representation of G inside the circle B. Let $seg(u)$ denote the segment representing vertex u in R. Then $start(u)$ is the endpoint of $seg(u)$ on B and $end(u)$ is the endpoint of $seg(u)$ inside B, see Figure 1. Let $arc(u,v)$ be the arc of B starting at $start(u)$ going in clockwise direction to $start(v)$ and $b(u,v)$ be the line segment $end(u), end(v)$. For nonadjacent vertices u and v, let $A(u,v)$ be the area bounded by $seg(u), arc(u,v), seg(v), b(u,v)$. Since $A(u,v)$ may not be a convex area, we need some extra definitions in order to capture a convex part of $A(u,v)$. Let $cord(u,v)$ be the unique cord of B which contains both $end(u)$ and $end(v)$. The line $cord(u,v)$ splits $A(u,v)$ into at most 3 convex parts. Let $A'(u,v)$ be the convex part having $b(u,v)$ as part of its border, see Figure 1 for an example of each of the three cases. For a line segment s, let $cross(s)$ denote the set of all vertices x such that $seg(x)$ intersects s and let $cross(u,v) = cross(seg(u)) \cup cross(b(u,v)) \cup cross(seg(v))$. We now define a function OS and show that this is a conflict-free function. For u not adjacent to v, let

$$OS(u,v) = \{x : end(x) \in A'(u,v) \text{ and } x \notin cross(u,v)\}$$

We define the reach extension of OS as ROS, c.f. Definition 2.1.

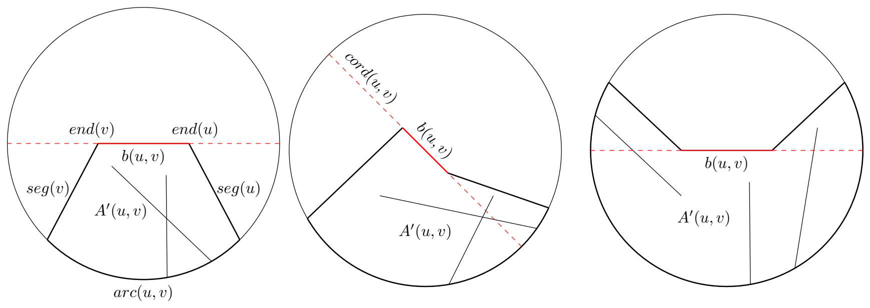

Figure 1: Three different cases for which we define A' with some notation marked on the figures. $b(u,v)$ and $chord(u,v)$ are drawn in red.

Lemma 2.7. *Let G be a graph and R be an outersegment representation of G. For every $u, v \in V(G)$, we have $\forall y \in ROS(u,v)$, the segment $seg(y)$ lies completely inside $A(u,v)$.*

Proof. For every $y \in ROS(u,v)$, either $y \in OS(u,v)$ or there exist $x \in OS(u,v)$ such that $y \in ROS(u,x) \cup ROS(x,v)$. For every $y \in OS(u,v)$ we have $end(y) \in A'(u,v) \subseteq A(u,v)$ and since $y \notin cross(u,v)$, we conclude that $seg(y)$ lies completely inside $A(u,v)$. For every $y \in ROS(u,v) \setminus SO(u,v)$ there must exist $x \in OS(u,v)$ such that $y \in ROS(u,x) \cup ROS(x,v)$. Since $A'(u,v)$ is convex and $end(x)$ lies in $A'(u,v)$, the triple $end(u), end(v), end(x)$ forms a triangle T and that none of the segments $seg(u), seg(v), seg(x)$ intersects T other than in the corners, this implies that $A(u,x) \cup A(x,v) \subseteq A(u,v)$, see Figure 2. We do induction on $|ROS(u,v)|$, the base cases are when $ROS(u,v) = \emptyset$ for which the lemma is trivially true. Assume the lemma true for every u', v' with $|ROS(u',v')| < |ROS(u,v)|$. Since $y \in ROS(u,x) \cup ROS(x,v)$ we have by assumption that $seg(y) \in A(u,x) \cup A(x,v)$ and we conclude that $seg(y)$ lies completely inside $A(u,v)$. By induction the lemma is true for all u, v, y. $\square$

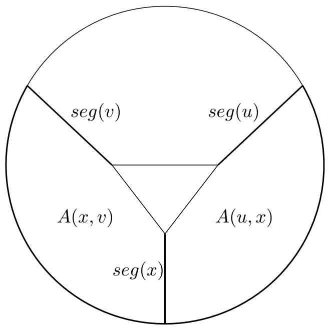

Figure 2: When updating the table, we choose an x to divide $A(u,v)$ into three parts $A(u,x), A(x,v)$ and a triangle. The main point of the proof of optimality is that for any independent set there exists an x in the independent set such that no segment belonging to a vertex in the independent set intersects with the triangle.

Lemma 2.8. *Let G be an outersegment graph and R be an outersegment representation of G. Then OS is a conflict-free function.*

Proof. Since neither u, nor v, nor any element of $cross(u, v)$ lies in $ROS(u, v)$, we have that $ROS(u, v) \cap N[u, v] = \emptyset$. Since $A(u, x)$ is disjoint from $A(x, v)$, by Lemma 2.7, $N[ROS(u, x)] \cap ROS(x, v) = \emptyset$. Hence OS satisfies the conditions of Definition 2.2. $\square$

This implies that Algorithm 1 returns the weight of an independent set. Now we will show how to ensure the returned solution is optimal.

Definition 2.9 (Normalized outersegment representation)**.** *Let G be an outersegment graph, R be an outersegment representation of G, and ROS be the reach function of OS. We define R as normalized if for all nonadjacent $u, v \in V(G)$, we have*

$$ROS(u, v) = \{x : seg(x) \subset A(u, v)\}.$$

We can modify any outersegment graph G with a representation R, with end points in general position, to become normalized by adding $3n$ isolated vertices of weight 0 and corresponding segments of length ϵ as described in Figure 3.

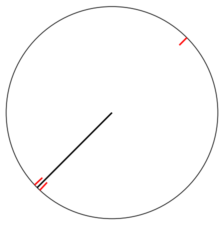

Figure 3: For every vertex v, add two segments starting at distance ϵ from $start(v)$, one on each side of $seg(v)$ and in additional, add one such segment where the extension of $seg(v)$ would cross the the circle B. The added segments are drawn in red.

Lemma 2.10. *The process described in Figure 3 will produce a normalized graph.*

Proof. Define $X(u, v) = \{x : seg(x) \subset A(u, v)\}$. It follows from Lemma 2.7 that for all nonadjacent $u, v \in V(G)$, we have $ROS(u, v) \subseteq X(u, v)$. Assume for contradiction that u, v is the pair with minimal $arc(u, v)$ such that there exists $y \in X(u, v) \setminus ROS(u, v)$. Since $y \notin ROS(u, v)$, we conclude that $y \notin OS(u, v)$ and hence $end(y) \in A(u, v) \setminus A'(u, v)$ implying that $A(u, v)$ is not convex. Thus we are in either the second or third case illustrated in Figure 1. If $start(y) \in A'(u, v)$, then there will be a segment y' of length ϵ with $start(y')$ at distance ϵ from $start(y)$ such that $end(y')$ is visible from both $end(u)$ and $end(v)$ in A', hence $y' \in OS(u, v)$ and $seg(y) \subset A(u, y')$ or $seg(y) \subset A(y', v)$. Then since $arc(u, v)$ is minimal $seg(y)$ is in either $ROS(u, y')$ or $ROS(y', v)$ contradicting that $y \notin ROS(u, v)$. If $start(y) \notin A'(u, v)$, then the whole segment $seg(y)$ must lie in $A(u, v) \setminus A'(u, v)$. Hence for one of the ϵ length segments x on the extensions of $seg(u)$ and $seg(v)$ will lie inside $A'(u, v)$ and we have $x \in OS(u, v) \setminus N(y)$. Then $seg(y)$ is in either $X(u, x) \setminus ROS(u, x)$ or $X(x, v) \setminus ROS(x, v)$ contradicting the minimality of $arc(u, v)$. $\square$

Theorem 2.11. *Given an outersegment graph G with n vertices and an outersegment representation R of G, the* Maximum Weight Independent Set *problem can be solved in $O(n^3)$ time.*

Proof. By Lemma 2.10, we can make G and R normalized in linear time and the graph still has $O(n)$ vertices. We assume that R has been normalized using the process described in Figure 3. By using Lemma 2.7, we can precompute $OS(u, v)$ and $ROS(u, v)$ for all pairs in $O(n^3)$ time since checking for segment intersection and containment in $A(u, v)$ or $A'(u, v)$ can be done in constant time. It follows from Lemma 2.8 and Theorem 2.3 that Algorithm 1, if called on G and OS, always returns the weight of an independent set and terminates in $O(n^3)$ time. We need to prove that for any independent set I, Algorithm 1 will return a weight that is at least $w(I)$. Note that it suffices to assume that all weights are non-negative and I is a maximal independent set.

We prove by induction on $|ROS(u,v)|$ that for all $u,v \in I$, the weight stored in $M[u,v]$ is at least the weight of $I \cap (ROS(u,v) \cup \{u,v\})$. The statement is trivially true for $ROS(u,v) = \emptyset$. Let $u,v \in I$ and assume the statement is true for all u', v', where $|ROS(u',v')| < |ROS(u,v)|$. There is a vertex $x \in I \cap OS(u,v)$ since $OS(u,v) = \emptyset \leftrightarrow ROS(u,v) = \emptyset$ and I is maximal. Let I_{uv} be $I \cap ROS(u,v)$. Since R has been normalized by the procedure of Figure 3, and I is maximal, I_{uv} will consist of groups of three segments, one original and two parallel of length ϵ, plus also singleton segments of length ϵ. Let P^I_{uv} be the simple polygon that starts at $end(v)$, then proceeds clockwise to $end(u)$ then to each of $end(s_i)$ where $i \in \{1,2,\ldots,k\}$ and s_i is the i-th clockwisemost segment in I_{uv}, and then back to $end(v)$, see Figure 4.

In a triangulation of $P^I_{u,v}$ there is a segment $seg(x)$ such that $end(x), end(u), end(v)$ forms a triangle which does not intersect $seg(u)$ nor $seg(v)$. We know that $end(x)$ must be in $A'(u,v)$ and hence $x \in OS(u,v)$. Since $x \in I$, we know that $N(x) \cap I = \emptyset$ and by the choice of x, we know that no vertex in $I \cap ROS(u,v)$ intersects with the triangle $end(u), end(x), end(v)$. Hence $I \cap (ROS(u,x) \cup ROS(x,v)) \cup \{u,x,v\} \supseteq I \cap ROS(u,v) \cup \{u,v\}$. Algorithm 1 will do the update

$$M[u,v] = max(M[u,v], M[u,x] + M[x,v] - w(x)).$$

By the induction hypothesis, $M[u,x] \geq w(I \cap ROS(u,x) \cup \{u,x\})$ and $M[x,v] \geq w(I \cap ROS(x,v) \cup \{x,v\})$ and thus $M[u,v] \geq w(I \cap (ROS(u,x) \cup ROS(x,v) \cup \{u,x,v\})$. By induction every entry of the table will be correct. $\square$

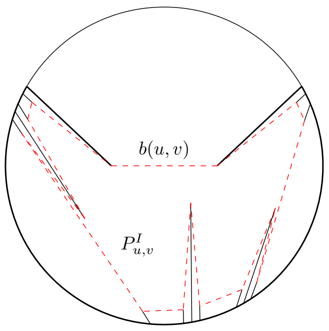

Figure 4: $I \cap ROS(u,v)$ drawn in black and $P^I_{u,v}$ drawn in dashed red.

2.2 Outerstring graphs

Outerstring graphs are intersection graphs of curves in the plane that lie inside a circle such that each curve intersects the boundary of the circle in one of its endpoints. An outerstring representation is an intersection model of non self-intersecting polygonal lines in the plane with one endpoint on a circle B, let N denote the total number of segments. There are outerstring graphs which require strings with exponential many segments in their outerstring representation [14].

Our runtime will be polynomial in the total number of segments, N, needed to represent the strings. We now adapt the OS function, defined for outersegment graphs, to work for outerstring graphs, the proofs are similar to those of Section 2.1 and we will redefine the notation from that section.

Let G be an outerstring graph and R be a representation as polygonal lines inside a circle B. Let $str(u)$ denote the string representing u in R, $start(u)$ denote the start-point of $str(u)$ on B, and $end(u)$ denote the end-point of $str(u)$ inside B. Let $b(u,v)$ be the straight line-segment $end(u), end(v)$. Let $arc(u,v)$ be the arc of B starting at $start(u)$ going in clockwise direction to $start(v)$. For a string s, let $cross(s)$ denote the set of all vertices x such that $str(x)$ intersects s and let $cross(u,v) = cross(str(u)) \cup cross(b(u,v)) \cup cross(str(v))$. Note that we may assume $u \notin cross(str(u))$, but might have $u \in cross(b(u,v))$. We say a set $S \subseteq V(G)$ is *open* if S is independent and $S \cap \bigcup_{u,v \in S} cross(b(u,v)) = \emptyset$. For all pairs u, v, such that $\{u,v\}$ is open, we define $A(u,v)$ be the area enclosed by $str(v), b(u,v), str(u)$, and $arc(u,v)$. For all pairs u, v, such that $\{u,v\}$ is not open, we define $OS(u,v) = \emptyset$; otherwise

$$OS(u,v) = \{x : start(x) \in arc(u,v) \text{ and } \{u,x,v\} \text{ is open } \}.$$

Let ROS be the reach extension of OS.

Lemma 2.12. *Let G be a graph and R be an outerstring representation of G, for every $u, v \in V(G)$, we have $\forall x \in ROS(u,v)$, the segment $str(x)$ will lie completely inside $A(u,v)$.*

Proof. For every $x \in OS(u,v)$ we have, $\{u,x,v\}$ is open, hence the triangle $end(u), end(x), end(v)$ does not intersect $str(u) \cup str(x) \cup str(v)$. Since $start(x) \in A(u,v)$ and $x \notin cross(u,v)$, we conclude that $str(x)$ will lie completely inside $A(u,v)$. This implies that $A(u,x) \cup A(x,v) \subseteq A(u,v)$, we conclude by same argument as for Lemma 2.7 that for every $x \in ROS(u,v)$, $seg(x)$ will lie completely inside $A(u,v)$. □

Lemma 2.13. *The function OS is conflict-free.*

Proof. From Lemma 2.12 we know that $OS(u,v)$ only contains vertices whose strings start on $arc(u,v)$ and lie entirely inside the area $A(u,v)$. This implies that $ROS(u,v) \cap N[u,v] = \emptyset$. For any $x \in OS(u,v)$ we have $A(u,x) \cap A(x,v) = \emptyset$. Since every string belonging to a vertex in $ROS(u,x)$ lies inside $A(u,x)$ and every string belonging to a vertex in $ROS(x,v)$ lies inside $A(x,v)$ we get that $N[ROS(u,x)] \cap ROS(x,v) = \emptyset$. Hence the conditions for being conflict-free are satisfied. □

This implies that calling Algorithm 1 on an outerstring graph will find an independent set, but not necessarily an optimal solution. We will now show how to draw an outerstring in a normalized form. We assume that the points in the outerstring representation are in general position, i.e. no three points on the bends of the polygonal paths are collinear.

Definition 2.14 (Normalized outerstring representation)**.** *Let G be an outerstring graph, R be an outerstring representation of G, and ROS be the reach extension of OS. We define R as normalized if for all nonadjacent $u, v \in V(G)$, we have*

$$ROS(u,v) = \{x : str(x) \subset A(u,v)\}.$$

We now describe a normalization procedure which is illustrated in Figure 5. Let G be a graph, and R be an intersection model of G by polygonal paths inside a circle B. For $u \in V(G)$, where $str(u) = (start(u) = b_u^0, b_u^1, b_u^2, \dots b_u^k = end(u))$ and for each bend $b_u^i, i = k-1, k-2, \dots 1$ add two vertices to G adjacent to all vertices whose corresponding strings intersect $str(u)$ closer to the start than b_u^i and set their weight to 0. Add two polygonal paths to R, one on each side of $str(u)$ at distance ϵ from the previously added polygonal path associated with u following $str(u)$ from $start(u)$ ending at b_i^i. Next, add two isolated vertices to G of weight 0 and corresponding segments to R of length ϵ at distance ϵ from the most recently added segments associated with u. Also, for each bend b_u^i of $u, i = 1 \dots k$, add a segment of length ϵ where the ray beginning at b_u^{i-1} containing b_u^i intersects the circle B. The graph now has $3N$ strings.

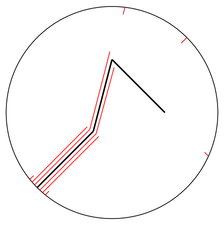

Figure 5: The result of the normalizing process, the added strings are drawn in red.

Lemma 2.15. *The process described above will produce a normalized outerstring graph.*

Proof. Let $X(u,v) = \{x : str(x) \subset A(u,v)\}$. Assume for contradiction that u, v is the pair with minimum length of $arc(u,v)$ such that $X(u,v) \setminus ROS(u,v) \neq \emptyset$. Let y be a vertex in $X(u,v) \setminus ROS(u,v)$. Since $y \notin ROS(u,v)$ also $y \notin OS(u,v)$. We will show that there exists a vertex $x \in OS(u,v)$ such that $y \in ROS(u,x) \cup ROS(x,v)$. The desired x has $start(x) \in arc(u,v)$ with $\{u,x,v\}$ open.

Let l_u and r_u be the points of length ϵ added close to $start(u)$ and p_u be the point where the ray beginning at b_u^{i-1} containing b_u^i intersects the circle B. We have $OS(u,v) \neq \emptyset$ since either some bend b_u^i of $str(u)$, l_y, r_y, p_u or p_v is visible from both u and v and hence the normalization has added a vertex which is in $OS(u,v)$.

We now describe a procedure to find x which we call the sweep procedure, refer to Figure 6. The idea is to sweep a maximal line segment s parallel to $b(u,v)$ such that $s \subset A(u,v) \setminus str(y)$. The endpoints of s may lie on $str(u)$, $str(v)$, $str(y)$ or circle B. Let s be at distance ϵ from $b(u,v)$. Now transform s by moving it away from $b(u,v)$ maintaining each endpoint of s in the segments or arc that presently contains it while keeping s parallel to $b(u,v)$. Let x be the first vertex such that $end(x) \in s$ and $\{u,v,y,x\}$ is an independent set.

If x is close to a bend or start point of $str(z)$ for some $z \in \{u,v,y\}$, then we know that $str(x)$ is tracing $str(z)$ within distance $f(\epsilon)$ of $str(z)$, since $\{u,v,y\}$ is an independent set we know that $\{u,v,y,x\}$ is an independent set.

At this point we have identified vertex x with $\{u,x,v\}$ open, such that $y \in ROS(u,z)$ or $y \in ROS(z,v)$. Since $ROS(u,v)$ contains $ROS(u,z) \cup ROS(z,v)$, this contradicts the minimality of $arc(u,v)$. □

Theorem 2.16. *Given an outerstring graph G with n vertices and an outerstring representaion R of G with N segments the* Maximum Weight Independent Set *problem can be solved in $O(N^4)$ time.*

Proof. By Lemma 2.15, we make G and R normalized in $O(N)$ time and the graph has $O(N)$ vertices. We can precompute $OS(u,v)$ and $ROS(u,v)$ for all pairs in $O(N^4)$ time by using the adjacency information of G and testing all triples u, x, v, since checking for containment in $A(u,v)$ can be done in $O(N)$ time. It follows from Lemma 2.13 and Theorem 2.3 that Algorithm 1, if called on G and OS, always returns the weight of an independent set and terminates in $O(N^3)$ time. We need to prove that for any independent set I, Algorithm 1 will return a weight that is at least $w(I)$. Note that it suffices to assume that I is a maximal independent set.

We prove by induction on $|ROS(u,v)|$ that for all $u, v \in I$, the weight stored in $M[u,v]$ is at least the weight of $I \cap (ROS(u,v) \cup \{u,v\})$. The statement is trivially true for $ROS(u,v) = \emptyset$ since this can only happen if $X(u,v) = \emptyset$. Let $u, v \in I$ and assume the statement is true for all u', v' where $|ROS(u',v')| < |ROS(u,v)|$. Since I is maximal and thus contains all ϵ length segments in $A(u,v)$, by a sweep procedure similar to that outlined in the proof of the previous lemma we can

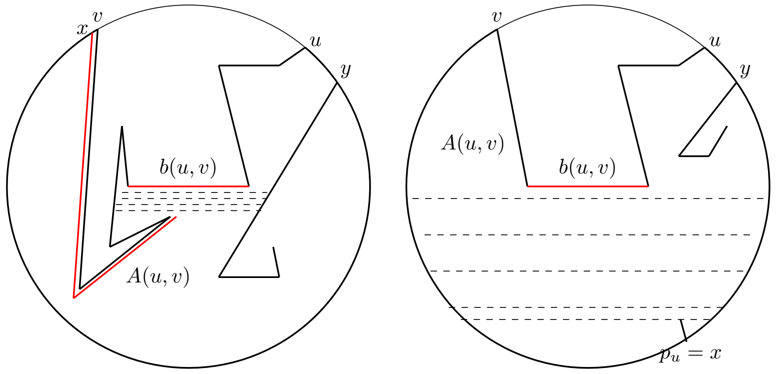

Figure 6: Sweeping s into $A(u,v)$ to find x

locate a vertex $x \in I \cap OS(u,v)$. Since $end(x), end(u), end(v)$ forms a triangle which is a subset of the area swept, the triangle does not intersect any string belonging to a vertex in I, hence we know that $I \cap (ROS(u,x) \cup ROS(x,v) \cup \{x\}) = I \cap ROS(u,v)$. Algorithm 1 will do the update

$$M[u,v] = max(M[u,v], M[u,x] + M[x,v] - w(x)).$$

By the induction hypothesis $M[u,x] \geq w(I \cap ROS(u,x) \cup \{u,x\})$ and $M[x,v] \geq w(I \cap ROS(x,v) \cup \{x,v\})$ and thus $M[u,v] \geq w(I \cap (ROS(u,x) \cup ROS(x,v) \cup \{u,x,v\})$. By induction every entry of the table will be correct. □

References

[1] Seymour Benzer. On the topology of genetic fine structure. *Proceedings of the National Academy of Sciences*, 47:1607, 1959.

[2] Sergio Cabello, Jean Cardinal, and Stefan Langerman. The clique problem in ray intersection graphs. In *ESA*, pages 241–252, 2012.

[3] Holger Flier, Matús Mihalák, Anita Schöbel, Peter Widmayer, and Anna Zych. Vertex disjoint paths for dispatching in railways. In *ATMOS*, pages 61–73, 2010.

[4] Holger Flier, Matús Mihalák, Peter Widmayer, and Anna Zych. Maximum independent set in 2-direction outersegment graphs. In *WG*, pages 155–166, 2011.

[5] Jacob Fox and János Pach. Computing the independence number of intersection graphs. In *SODA*, pages 1161–1165, 2011.

[6] Jacob Fox and János Pach. Coloring K_k-free intersection graphs of geometric objects in the plane. *Eur. J. Comb.*, 33(5):853–866, 2012.

[7] Jacob Fox and János Pach. String graphs and incomparability graphs. *Advances in Mathematics*, 230:1381–1401, 2012.

[8] Takeshi Fukuda, Yasuhiko Morimoto, Shinichi Morishita, and Takeshi Tokuyama. Data mining with optimized two-dimensional association rules. *ACM Transactions on Database Systems*, 26(2):179–213.

[9] Fanica Gavril. Maximum weight independent sets and cliques in intersection graphs of filaments. *Information Processing Letters*, 73:181–188, 2000.

[10] Jan Kratochvíl. String graphs. I. the number of critical nonstring graphs is infinite. *J. Comb. Theory, Ser. B*, 52(1):53–66, 1991.

[11] Jan Kratochvíl. String graphs. II. recognizing string graphs is NP-hard. *J. Comb. Theory, Ser. B*, 52(1):67–78, 1991.

[12] Jan Kratochvíl, Miroslav Goljan, and Petr Kučera. *String Graphs.* Academia, Prague, 1986.

[13] Jan Kratochvíl, Anna Lubiw, and Jaroslav Nesetril. Noncrossing subgraphs in topological layouts. *SIAM J. Discrete Math.*, 4(2):223–244, 1991.

[14] Jan Kratochvíl and Jirí Matousek. String graphs requiring exponential representations. *J. Comb. Theory, Ser. B*, 53(1):1–4, 1991.

[15] Jan Kratochvíl and Jaroslav Nesetril. Independent set and clique problems in intersection-defined classes of graphs. *Comment. Mat. Univ. Carolinae*, 31:85–93, 1990.

[16] Doron Rotem Martin Golumbic and Jorge Urrutia. Comparability graphs and intersection graphs. *Discrete Mathematics*, 43:37–46, 1983.

[17] M. Middendorf and F. Pfeiffer. The max clique problem in classes of string-graphs. *Discrete Mathematics*, 108:365–372, 1992.

[18] Marc van Kreveld Pankaj Agarwal and Subhash Suri. Label placement by maximum independent set in rectangles. *Computational Geometry: Theory and Applications*, 11(3):209–218.

[19] S. Even S. Ehrlich and R.E. Tarjan. Intersection graphs of curves in the plane. *J. Combin. Theory Ser. B.*, 21:8–20, 1976.

[20] Marcus Schaefer, Eric Sedgwick, and Daniel Stefankovic. Recognizing string graphs is in NP. *J. Comput. Syst. Sci.*, 67(2):365–380, 2003.

[21] F.W. Sinden. Topology of thin film rc-circuits. *Bell System Technological Journal*, 45:1639–1662, 1966.

A Extra Figures

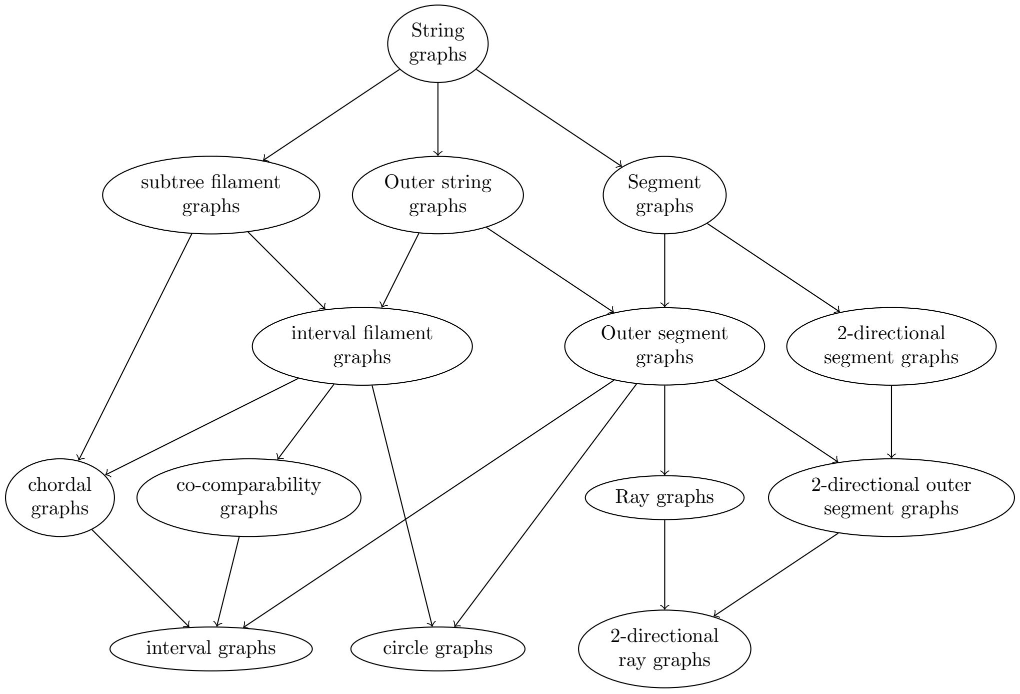

Figure 7: Containment relationships between various graph classes referred to in this paper

B Proof of Theorem 2.5

Lemma B.1. *Let $u, v \in V(G)$, then for any σ, we have $S_\sigma(u,v) \cap N[u,v] = \emptyset$.*

Proof. Assume for contradiction that there is a vertex $x \in S_\sigma(u,v) \cap N[u,v]$. Then by the definition of $S_\sigma(u,v)$ (the strict inequalities) $x \notin \{u,v\}$. Hence we must have $x \in N(u,v)$. Without loss of generality, assume that $u \in N(x)$. Then $N(x) \setminus N(u,v)$ will contain u and the condition will fail for $y = u$. □

Lemma B.2. *Let G be a graph and $\{u,x,v\} \subseteq V(G)$ be an independent set. Let σ be an ordering of $V(G)$ such that $\sigma(u) < \sigma(x) < \sigma(v)$. Then there is no edge $(a,b) \in E(G)$ such that, $a \in S_\sigma(u,x)$ and $b \in S_\sigma(x,v)$.*

Proof. Assume for contradiction that $a \in S_\sigma(u,x)$, $b \in S_\sigma(x,v)$ and $(a,b) \in E(G)$. By Lemma B.1, we get that $b \notin N[x,v]$ and neither can we have $b \in N(u)$. Since $b \in S_\sigma(x,v)$ implies $\sigma(x) < \sigma(b)$, $b \in (N(a) \setminus N(u,x))$ but $(x,b) \notin E$. The condition that $a \in S_\sigma(u,x)$ when $y = b$ gives that $\sigma(u) < \sigma(b) < \sigma(x)$ but this contradicts $b \in S_\sigma(x,v)$. □

Lemma B.3. *Let G be a graph and $u, v \in V(G)$. If $z \in S_\sigma(u,v)$, then $S_\sigma(u,z) \subset S_\sigma(u,v)$.*

Proof. Assume for contradiction that $\exists x \in (S_\sigma(u,z) \setminus S_\sigma(u,v))$. Since $z \in S_\sigma(u,v)$ and $x \in S_{u,z}$ we have $\sigma(u) < \sigma(x) < \sigma(z) < \sigma(v)$ and by definition

$$\forall y \in (N(z) \setminus N(u,v) \cup \{z\}), \sigma(u) < \sigma(y) < \sigma(v) \text{ and}$$

$$\forall y \in (N(x) \setminus N(u,z) \cup \{x\}), \sigma(u) < \sigma(y) < \sigma(z) < \sigma(v)$$

We see that the set $(N(z) \setminus N(u,v) \cup \{z\}) \cup (N(x) \setminus N(u,z) \cup \{x\})$ contains $N(x) \setminus N(u,v)$, hence by combining the two above statements we get

$$\forall y \in (N(x) \setminus N(u,v) \cup \{x\}), \sigma(u) < \sigma(y) < \sigma(v)$$

Contradicting $x \in (S_\sigma(u,z) \setminus S_\sigma(u,v))$. □

Theorem 2.5 *For all graphs G and all orderings σ of $V(G)$, the function $S_\sigma(u,v)$ is conflict-free.*

Proof. Given σ, let S be S_σ. First we show that the reach extension RS of S is identical to the function S. By definition $S(u,v) \subseteq RS(u,v)$. We show that $RS(u,v) = S(u,v)$ by induction on the size of $S(u,v)$. If $S(u,v) = \emptyset$, then $RS(u,v) = S(u,v)$ by definition. From Lemma B.3, we get that $\forall x \in S(u,v) : S(u,x) \cup S(x,v) \subset S(u,v)$, hence by induction hypothesis $\forall x \in S(u,v) : S(u,x) \cup S(x,v) = RS(u,x) \cup RS(x,v)$. Combining this with Definition 2.1, we get

$$RS(u,v) = S(u,v) \cup \bigcup_{x \in S(u,v)} (S(u,x) \cup S(x,v)) = S(u,v).$$

The theorem now follows from Lemma B.1, B.2 and B.3. □

Lemma B.4. *Given G and σ, we can compute $S_\sigma(u,v)$ for all pairs $u,v \in V(G)$ in $O(n^3 + n^2m)$ time.*

Proof. Let $u,v \in V$ with $\sigma(u) < \sigma(v)$. By Lemma B.1, $u,v \in S_\sigma(u,v)$. Let $x \in V$ and $\sigma(u) < \sigma(x) < \sigma(v)$. We need to check for all $y \in (N(x) \setminus N(u,v)) \cup \{x\}$, $\sigma(u) < \sigma(y) < \sigma(v)$. This can be done by taking a neighbor y of x and checking whether y is a neighbor of u or neighbor of v. The checking of whether y is a neighbor of u or v can be done in constant time by looking up in the adjacency matrix which can be computed in $O(n^2)$ time. If y is either a neighbor of u or v, then discard x; otherwise include x to $S_\sigma(u,v)$. Again since the choice of x can be at most n, the computation of $S_\sigma(u,v)$ takes at most $O(n+m)$ time. There are at most n^2 pairs of such u and v. So the computation of $S_\sigma(u,v)$ for all pairs u and v in G takes at most $O(n^2(n+m)) = O(n^3+n^2m)$ time. □