

## Egalitarian Graph Orientations

Glencora Borradaile<sup>1</sup> Jennifer Iglesias<sup>1</sup> Theresa Migler<sup>1</sup> Antonio  
Ochoa<sup>1</sup> Gordon Wilfong<sup>2</sup> Lisa Zhang<sup>2</sup>

<sup>1</sup>Oregon State University, Corvallis, OR

<sup>2</sup>Nokia Bell Labs, Murray Hill, NJ

### Abstract

Given an undirected graph, one can assign directions to each of the edges of the graph, thus *orienting* the graph. To be as egalitarian as possible, one may wish to find an orientation such that no vertex is unfairly hit with too many arcs directed into it. We discuss how this objective arises in problems resulting from telecommunications. We give optimal, polynomial-time algorithms for: finding an orientation that minimizes the lexicographic order of the indegrees and finding a strongly-connected orientation that minimizes the maximum indegree. We show that minimizing the lexicographic order of the indegrees is NP-hard when the resulting orientation is required to be acyclic.

Submitted: July 2016	Reviewed: October 2016	Revised: December 2016	Reviewed: March 2017	Revised: May 2017
	Accepted: May 2017	Final: May 2017	Published: June 2017	
	Article type: Regular paper		Communicated by: A. Lubiw	

Research supported by NSF CCF-0963921 and NSF 0852030.

*E-mail addresses:* glencora@eecs.orst.edu (Glencora Borradaile) jiglesia@andrew.cmu.edu (Jennifer Iglesias) tmigler@calpoly.edu (Theresa Migler) adjchoa@gmail.com (Antonio Ochoa) gordon.wilfong@nokia-bell-labs.com (Gordon Wilfong) lisa.zhang@nokia-bell-labs.com (Lisa Zhang)

## 1 Introduction

We consider problems of orienting the edges of an undirected graph so that no vertex is unfairly hit with too many arcs directed into it. We refer to such orientations as *egalitarian*: the total available indegree is shared among the vertices as equally as allowed by the topology of the graph. This objective arises in various telecommunications problems. Depending on the requirements of the problem, the orientation may be unconstrained or need to be strongly connected or acyclic. We start by describing these motivating applications and related work.

**Unconstrained orientations** Venkateswaran introduced the problem of directing the edges of an undirected graph so as to minimize the maximum indegree [19]. The problem arises from a telecommunications network design problem in which source-sink pairs  $(s_i, t_i)$  are linked by a directed  $s_i$ -to- $t_i$  path  $c_i$  (called a *circuit*). When an edge of the network fails, all circuits using that edge fail and must be rerouted. For each failed circuit, the responsibility for finding an alternate path is assigned to either the source or sink corresponding to that circuit. To limit the rerouting load of any vertex, it is desirable to minimize the maximum number of circuits for which any vertex is responsible.

Venkateswaran models this problem with an undirected graph whose vertices are the sources and sinks and whose edges are the circuits. He assigns the responsibility of a circuit's potential failure by orienting the edge to either the source or the sink of this circuit. Minimizing the maximum number of circuits for which any vertex is responsible can thus be achieved by finding an orientation that minimizes the maximum indegree of any vertex. Venkateswaran shows how to find such an orientation [19]. Asahiro, Miyano, Ono, and Zenmyo give a simpler analysis [5]. Asahiro et al show further that for any  $w \geq \lceil \frac{\text{maxdegree}}{2} \rceil$  the PATH REVERSAL algorithm maximizes the number of vertices with indegree at most  $w$  and consequently minimizes the number of vertices with indegree at least  $w + 1$  [3].

However, there may be multiple orientations that have the same minimized maximum indegree. The orientation that has the minimum number of vertices with maximum indegree is preferable since it minimizes the number of vertices that have the maximum rerouting load. Among the orientations that minimize the number of vertices with maximum indegree, the one that minimizes the number of second largest indegree is preferable for the same reason of rerouting load. Continuing this reasoning, we can formalize this notion in the following way: given two orientations  $G_A$  and  $G_B$ , we prefer  $G_A$  to  $G_B$  if the sequence of indegrees of  $G_A$  (in non-increasing order) is lexicographically before the sequence of indegrees of  $G_B$  (in non-increasing order). We refer to finding the best orientation with respect to this measure as the *minimum lexicographic* orientation. In Section 2.1, we show that a natural greedy algorithm finds the minimum lexicographic orientation: start with an arbitrary orientation and repeatedly reverse the orientation of a directed path while doing so improves the objective.

**Strongly-connected orientations** Networks that are used to route messages should, naturally, be strongly connected: one should be able to send a packet along a directed path from any vertex to any other vertex. A destination-based routing protocol chooses the next arc along which to send a message based only on the destination of the message. Such a protocol can be implemented with an *interval routing scheme* [16]. An interval routing scheme for a directed graph is defined by a cyclic numbering of the vertices and a labeling of each arc with an interval of the vertex numbers. (More generally, each edge can be labelled with multiple intervals. We will show that one interval is sufficient and therefore the best possible.) For each vertex  $u$ , the disjoint union of the intervals labeling the outgoing arcs from  $u$  cover all but  $u$ 's vertex numbers. When a packet destined for vertex  $v$  reaches a vertex  $u \neq v$  it is forwarded from  $u$  along the outgoing arc from  $u$  whose label contains the interval containing  $v$ 's number. Such a scheme, in order to be feasible, must be such that a packet originating at any vertex destined for any other vertex will reach the destination vertex when routing is done as described above.

In Section 3, we show that for any strongly connected graph there is an interval routing scheme such that each outgoing arc is labelled with at most one interval. This is the most compact routing scheme possible and allows the routing decision at a given vertex to be made in time proportional to the outdegree of that vertex. Thus, to minimize the routing time at each vertex, we would like to find a strongly-connected orientation of  $G$ , the underlying physical network, which minimizes the maximum number of outgoing arcs from any vertex. To keep the notation the same between sections of this paper, we instead minimize the maximum indegree; this is equivalent by way of reversing all the edges of the graph. We give an algorithm to find such an orientation in Section 3.2. We conjecture that the natural generalization of this algorithm also finds the minimum lexicographic order of the indegrees of the graph.

Santoro and Khatib introduced interval routing [18], and Van Leeuwen and Tan extended the idea [16]. Since then, there has been a large body of work on the topic. For a comprehensive survey, please see [13]. This survey presents various algorithms for the problems, depending on whether linear or cyclic intervals are used, how well the algorithms approximate shortest path routing and the level of compactness which is the maximum number of intervals on an edge. The work discussed in the survey focuses on undirected graphs, and as far as we know our work on directed graphs is new. Unfortunately, in the case we consider, the paths produced by interval routing can be a factor of  $n$  longer than the shortest paths in general directed graphs such as a ring with  $n$  vertices.

**Acyclic orientations** Consider a packet network with input buffers. A vertex can forward a packet from its input buffer to the next-hop (the next vertex in the packet's route) if the input buffer of the next-hop is not already full. Such networks can suffer from deadlock. For example, consider a ring network in which all input buffers are full: no vertex can forward a packet to its next-hop because the next-hop's input buffer is full. If no packet is allowed to go along

certain length-two paths then deadlock is prevented. In particular, Wittorff shows how to find such a collection of forbidden length-two paths by orienting the edges of the network so that the resulting graph is acyclic with a single source and making every pair of edges oriented into the same vertex a forbidden length-two path [22]. Then a path between every pair of vertices avoiding forbidden paths can be found that avoids any transition from travelling along an arc to travelling along the reverse of another arc (and hence avoids a pair of edges that get directed into the same vertex). Minimizing the maximum indegree minimizes the number of forbidden pairs at any vertex and hence minimizes the number of routing constraints at any vertex.

In Section 4 we present a simple algorithm to find an acyclic orientation for the objective of minimizing the maximum indegree. On the other hand, we also show that minimizing the lexicographic order of the indegrees is NP-hard when the resulting orientation must be acyclic.

## 1.1 Related work

Asahiro et al. consider the edge-weighted version of the unconstrained problem [5]. They build on the work of Venkateswaran and give a  $2 - 1/k$ -approximation algorithm where  $k$  is the maximum weight of any edge in the graph. They further show that the weighted version of the problem is strongly NP-hard even if all edge weights belong to the set  $\{1, k\}$  where  $k \geq 2$  is an integer [4]. Klostermeyer considers the problem of reorienting edges (rather than whole paths) so as to create graphs with given properties, such as strongly connected graphs and acyclic graphs [15]. De Fraysseix and de Mendez show that they can find an indegree assignment of the vertices given a particular property [11]. In our work we are searching for a particular degree assignment not known a priori.

Biedl, Chan, Ganjali, Hajiaghayi, and Wood give a  $\frac{13}{8}$ -approximation algorithm for finding an ordering of the vertices such that for each vertex  $v$ , the neighbors of  $v$  are as evenly distributed to the right and left of  $v$  as possible [6]. For the purpose of a deadlock prevention problem described by Wimmer in [21], Wittorff describes a heuristic for finding an acyclic orientation that minimizes the sum over all vertices of the function  $\delta(v)$  choose 2, where  $\delta(v)$  is the indegree of vertex  $v$  [22].

## 1.2 Notation

We use basic notation for graph theoretic concepts for graphs  $G = (V, E)$  with  $n$  vertices and  $m$  edges. A directed edge or arc,  $a$ , is oriented from the vertex  $\text{tail}(a)$  to the vertex  $\text{head}(a)$ . For a directed graph, the *indegree* of a vertex  $v$ , denoted  $\delta(v)$ , is the number of arcs for which  $v$  is the head. We may use a subscript to denote the graph with respect to which we measure the degree. A *directed path* is a sequence of arcs  $a_1, a_2, \dots, a_k$  with  $\text{head}(a_i) = \text{tail}(a_{i+1})$  for  $1 \leq i < k$ . We add trivial paths to this definition which are identified by a single vertex. A cycle is a path such that  $\text{head}(a_k) = \text{tail}(a_1)$ . An *orientation* of

an undirected graph is an assignment of directions to each edge in the graph. A directed graph is *strongly connected* if for every pair of vertices,  $u, v \in V$ , there are directed paths from  $u$  to  $v$  and from  $v$  to  $u$ . A directed graph is *acyclic* if there are no directed cycles in the graph. For a subset of vertices  $X$ ,  $G[X]$  is the subgraph induced by  $X$  and  $m(X)$  is the number of edges in  $G[X]$ .

## 2 Unconstrained orientations

We will show that a simple, greedy algorithm, first given by de Fraysseix and de Mendez [11], finds an orientation of an undirected graph that minimizes the lexicographic order of the indegrees. We say that a directed path from  $u$  to  $v$  is *reversible* if  $\delta(u) < \delta(v) - 1$ . The greedy algorithm, given an undirected graph, is:

PATH-REVERSAL

arbitrarily orient every edge  
 while there is a reversible path  
     let  $P$  be any reversible path whose last vertex is of highest indegree  
     reverse the orientation of each arc of  $P$

This algorithm can be implemented in quadratic time by arguing that the algorithm proceeds in  $k$  phases where  $k$  is the maximum indegree in the initial orientation (below). Therefore, there are at most  $m$  iterations of the algorithm and each iteration can be implemented in linear time using, for example, depth-first search.

Consider any integer  $\ell \leq k$ . Consider an iteration in which we reverse a  $u$ -to- $v$  path where  $\delta(v) = \ell$ . Let  $Q$  be the set of vertices of indegree  $> \ell$  just before this reversal and let  $Q'$  be the set of vertices that have paths to a vertex in  $Q$ . (Note:  $Q \subseteq Q'$ .) By definition neither  $u$  nor  $v$  is in  $Q'$ , for otherwise, we would reverse a path ending in a vertex of indegree  $> \ell$ . Further, after this reversal,  $Q$  is still the set of vertices of indegree  $> \ell$  and  $Q'$  is still the set of vertices that have paths to a vertex in  $Q$ . It follows that there is a well-defined phase  $\ell$ , a contiguous subset of iterations that reverse paths ending in vertices of indegree  $\ell$ : after reversing a path ending in a vertex of indegree  $\ell$ , the algorithm does not reverse a path ending in a vertex of higher indegree.

### 2.1 Minimizing the lexicographic order

PATH-REVERSAL finds an orientation that minimizes the maximum indegree. This observation was made by Venkateswaran with a rather involved proof [19]; a simpler analysis was given by Asahiro et al. [5]. This observation is also implied by de Fraysseix and de Mendez, Lemma 1 [11].

PATH-REVERSAL is more powerful than simply minimizing the maximum indegree. We show that the resulting orientation, in fact, minimizes the lexicographic order of the indegrees.

We define a *cycle reversal* to be the reversal of every edge in a cycle. Notice that performing a cycle reversal will not change the number of vertices of any particular indegree.

**Lemma 1** *Let  $O_1$  and  $O_2$  be orientations such that  $\delta_{O_1}(v) = \delta_{O_2}(v)$  for all  $v$ . Then  $O_1$  can be transformed into  $O_2$  by a sequence of cycle reversals.*

**Proof:** Let  $E_R$  be the set of arcs in  $O_1$  that have an opposite orientation in  $O_2$ . Notice that for all vertices  $v$  in the graph induced by  $E_R$ , the indegree of  $v$  equals the outdegree of  $v$ . For otherwise, there is some  $v$  such that  $\delta_{O_1}(v) \neq \delta_{O_2}(v)$ . It follows that each connected component of the graph induced by these edges has an Euler tour [14]. Reversing these tours gives the lemma.  $\square$

We define a *weak reversal* to be the reversal of a path from a vertex  $u$  to a vertex  $v$  where  $\delta(u) = \delta(v) - 1$ . Notice that performing a weak reversal will not change the number of vertices of any particular indegree.

**Theorem 1** *Any orientation that minimizes the lexicographic order of the indegrees of the vertices can be transformed into an orientation induced by PATH-REVERSAL via a sequence of weak reversals or cycle reversals.*

**Proof:** Let  $D_{lex}$  denote an orientation that minimizes the lexicographic order of the indegrees of the vertices, and let  $D_{PR}$  denote an orientation given by PATH-REVERSAL. Let  $\delta_{lex}(v)$  and  $\delta_{PR}(v)$  be the indegree of a vertex  $v$  in  $D_{lex}$  and  $D_{PR}$  respectively.

We will use induction on  $S := \sum_{v \in V} |\delta_{lex}(v) - \delta_{PR}(v)|$ . If  $|S| = 0$  then by Lemma 1, the theorem holds. Now suppose that  $S > 0$ . Let  $S_{\neq} = \{v : \delta_{lex}(v) \neq \delta_{PR}(v)\}$ . Let  $v$  be a vertex in  $S_{\neq}$  that maximizes  $\delta_{lex}(v)$  and if there is a choice among many such vertices, then maximizes  $\delta_{PR}(v)$ . Then we have the following two cases:

1.  $\delta_{lex}(v) > \delta_{PR}(v)$ . Let  $U$  be the set of all vertices that can reach  $v$  in  $D_{lex}$ . Notice that

$$\sum_{u \in U} \delta_{lex}(u) \leq \sum_{u \in U} \delta_{PR}(u) \quad (1)$$

This is because  $\sum_{u \in U} \delta_{lex}(u)$  is the number of edges in  $G[U]$ ,  $\sum_{u \in U} \delta_{PR}(u)$  also includes the indegree from edges in  $U$  and may additionally include the indegree from edges directed into  $U$ .  $\delta_{lex}(v) > \delta_{PR}(v)$  and  $v \in U$ , so there must be some  $u \in U$  with  $\delta_{lex}(u) < \delta_{PR}(u)$ . Because we chose  $v$  to maximize  $\delta_{lex}(v)$ ,  $\delta_{lex}(u) \leq \delta_{lex}(v)$ . Furthermore,  $\delta_{lex}(u) \neq \delta_{lex}(v)$  because if  $\delta_{lex}(u) = \delta_{lex}(v)$  then  $\delta_{PR}(u) > \delta_{lex}(u) = \delta_{lex}(v) > \delta_{PR}(v)$ , but we chose  $v$  to maximize  $\delta_{PR}(v)$ . Therefore  $\delta_{lex}(u) < \delta_{lex}(v)$ . It is not possible for  $\delta_{lex}(u) < \delta_{lex}(v) - 1$  for otherwise reversing a  $u$  to  $v$  path would give an orientation with a smaller lexicographic order than  $D_{lex}$ . Therefore

we have that  $\delta_{lex}(u) = \delta_{lex}(v) - 1$  and there is a weakly reversible path from  $u$  to  $v$  in the directed graph defined by  $D_{lex}$ . Reversing this path decreases  $S$  by 2.

2.  $\delta_{lex}(v) < \delta_{PR}(v)$ . Let  $U$  be the set of vertices that reach  $v$  in  $D_{PR}$ . Notice that  $\sum_{u \in U} \delta_{lex}(u) \geq \sum_{u \in U} \delta_{PR}(u)$ .  $\delta_{lex}(v) < \delta_{PR}(v)$  and  $v \in U$ , so there must be some  $u \in U$  with  $\delta_{lex}(u) > \delta_{PR}(u)$ . We chose  $v$  to be a vertex that maximizes  $\delta_{lex}(v)$ , so  $\delta_{lex}(u) \leq \delta_{lex}(v)$ . So we have  $\delta_{PR}(u) < \delta_{lex}(u) \leq \delta_{lex}(v) < \delta_{PR}(v)$ . This means that there is a reversible path from  $u$  to  $v$ , a contradiction because  $D_{PR}$  has no reversible paths.

□

**Corollary 2** *The algorithm PATH-REVERSAL finds an orientation that minimizes the lexicographic order of the indegrees.*

**Remarks** Let us revisit the motivating problem of failure recovery in network design, for which a failed circuit notifies either its source vertex or its sink vertex. We argued that minimizing the maximum indegree or the lexicographic order of the indegrees minimizes effort in failure recovery. We could also measure the recovery effort per vertex as a function  $f(\cdot)$  of the number of circuits this vertex is responsible for. The total effort for error recovery is then  $\sum_v f(\delta(v))$ . The shape of  $f(\cdot)$ , convex or concave or other more complex nature, can be debated. However, if  $f(\cdot)$  is increasing and strictly convex, we remark that the algorithm PATH-REVERSAL as we have seen also minimizes the total effort. Asahiro et al. [2] present a network flow algorithm that also gives an orientation that minimizes  $\sum_v f(\delta^+(v))$  where  $\delta^+(v)$  denotes the outdegree of a vertex  $v$  when  $f$  is convex.

**Theorem 3** *The algorithm PATH-REVERSAL finds an orientation  $G$  that minimizes  $F(G) = \sum_v f(\delta_G(v))$  for any increasing and strictly convex function  $f$ .*

**Proof:** Let  $\alpha_i(G)$  denote the number of vertices of indegree  $i$  in  $G$ . We rewrite the objective to be  $F(G) = \sum_i \alpha_i(G) \cdot f(i)$ .

Let  $G_a$  be an orientation of the graph that minimizes the given objective. Let  $G_b$  be the result of the algorithm PATH-REVERSAL using  $G_a$  as the initial orientation. By Theorem 2,  $G_b$  minimizes the lexicographic order of the indegrees. Since the non-increasing sequence of indegrees that corresponds to such an orientation is unique, it follows that any orientation  $G_c$  that minimizes the lexicographic order of the indegrees satisfies  $\alpha_i(G_c) = \alpha_i(G_b)$  for all  $i$ . It further follows that all orientations that minimize the lexicographic order of the indegrees achieve the same objective:  $F(G_c) = F(G_b)$ .

Suppose for a contradiction to the theorem that the degree distributions of  $G_b$  and  $G_a$  differ. Therefore, the algorithm PATH-REVERSAL performs at least one path reversal. Let  $G_1$  be the graph obtained from  $G_a$  after reversing one path, say from a vertex  $u$  to a vertex  $v$ .

We compare  $F(G_a)$  to  $F(G_1)$ . Let  $\delta_{G_a}(u) = k$  and  $\delta_{G_a}(v) = \ell$ . Since this path from  $u$  to  $v$  was a reversible path in  $G_a$ ,  $k < \ell - 1$ . By the path reversal operation, we get  $\delta_{G_1}(u) = k + 1$  and  $\delta_{G_1}(v) = \ell - 1$ . Then:

$$\begin{aligned} F(G_a) - F(G_1) &= \sum_x f(\delta_{G_a}(x)) - \sum_x f(\delta_{G_1}(x)) \\ &= f(\delta_{G_a}(v)) - f(\delta_{G_1}(v)) + f(\delta_{G_a}(u)) - f(\delta_{G_1}(u)) \\ &= \underbrace{f(\ell) - f(\ell - 1)}_A - \underbrace{(f(k + 1) - f(k))}_B \end{aligned}$$

Since  $k < \ell - 1$  and  $f$  is increasing and strictly convex, term  $A$  is greater than term  $B$ , and so the above difference is positive. It follows that  $F(G_a) > F(G_1)$ , contradicting the fact that  $G_a$  minimizes the objective  $F(G)$ . Therefore,  $G_a$  and  $G_1$  must have the same degree distribution.  $\square$

### 3 Strongly connected orientations

In this section we will show how to find a strongly-connected orientation that minimizes the maximum indegree (Section 3.2). We argue that this would enable an interval routing scheme (as described in the introduction) with minimum table sizes (Section 3.1). A routing table for a vertex  $v$  assigns intervals to each outgoing arc that encodes how a message should leave  $v$ . The size of a table for a given vertex  $v$  is the number of intervals summed over all outgoing arcs from  $v$ . Our routing scheme assigns exactly one interval to each arc (Theorem 5), therefore finding a strongly connected orientation of an undirected graph that minimizes the maximum *outdegree* of the vertices enables interval routing with minimum table sizes. Note that we can minimize the maximum outdegree by taking the reverse of an orientation that minimizes the maximum indegree. (Our orientation minimizes indegree rather than outdegree directly in keeping with the notation of the rest of this paper and the literature on such orientation problems.)

The construction in Figure 1 demonstrates that the maximum outdegree can be reduced significantly while keeping the orientation strongly connected, illustrating that minimizing the maximum outdegree can change the size of routing tables significantly.

#### 3.1 Minimum routing tables for strongly connected graphs

It is well known, as a generalization of Whitney's characterizations of 2-edge connected, undirected graphs [20] and Robbins' correspondence between strong connectivity and 2-edge connectivity [17], that a directed graph is strongly connected if and only if it has an ear-decomposition. An *ear decomposition* of a directed graph is a partition of the edges into a simple directed cycle  $P_0$  and

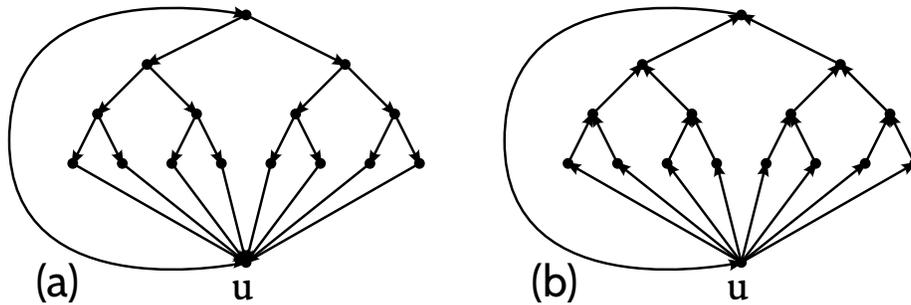


Figure 1:  $G_k$ ,  $k > 0$ , is a family of graphs composed of a perfect binary tree  $T_k$  with  $k + 1$  levels of vertices plus an additional vertex  $u$  that is connected to the root and every leaf of  $T_k$ . Consider two strongly connected orientations of  $G_k$  (illustrated for  $k = 3$ ): (a) Edges of  $T_k$  are oriented from leaf-to-root. Edges are oriented from the root of  $T_k$  to  $u$  and from  $u$  to the leaves of  $T_k$ . In this orientation, every vertex has outdegree 1 except for  $u$  which has outdegree  $2^k$ , the number of leaves in  $T_k$ . (b) Edges of  $T_k$  are oriented from root-to-leaf. Edges are oriented from  $u$  to the root of  $T_k$  and from the leaves of  $T_k$  to  $u$ . In this orientation, the leaves of  $T_k$  and  $u$  have outdegree 1 and every other vertex has outdegree 2.

simple directed paths (or cycles)  $P_1, \dots, P_k$  such that for each  $i > 0$ , the intersection of  $P_i$  with  $\cup_{j < i} P_j$  are the endpoints of  $P_i$  (which may be coincident if  $P_i$  is a cycle). Each  $P_i$  is called an ear.

An ear decomposition can be found in linear time [7]. Given an ear decomposition of a strongly-connected graph, we can define the routing tables using the procedure ROUTING below. We will define a cyclic ordering  $\mathcal{L}$  of the vertices. For each arc  $uv$ , we will define an interval  $[a, b]$ ,  $a, b \in \mathcal{L}$ . Recall from the introduction that this information can be used for routing: a message at a vertex  $u$  with destination  $d$  will be forwarded along  $uv$  if  $d$  is in  $[a, b]$ , that is if  $d$  is (inclusively) between  $a$  and  $b$  in the cyclic ordering  $\mathcal{L}$ . We say that such a labeling is feasible if it allows a message to be routed between any pair of vertices.

We assume, without loss of generality, that each ear in the ear decomposition contains at least two edges: a single-edge ear could be removed while maintaining strong connectivity and so will not be required for routing. We denote the number of edges in  $P$  by  $|P|$ . See Figure 2 for a demonstration of this procedure. It is convenient to use both open and closed endpoints for intervals of  $\mathcal{L}$ . For example,  $(a, b]$  contains all the vertices that are strictly after  $a$  and before (or equal to)  $b$  in the ordering. We use  $(a, a)$  to represent all the vertices in the cyclic ordering except  $a$ . Further, for the purposes of analysis, it is convenient to think of the intervals as being continuous.

ROUTING (ear decomposition  $P_0, P_1, \dots, P_k$ )

Initialize  $\mathcal{L}$  to contain all the vertices of  $P_0$  in their order around  $P_0$ .

Assign each arc  $ab$  of  $P_0$  the interval  $(a, a)$ .  
 For  $i = 1, \dots, k$ :  
     Let  $v_1$  be the first vertex of  $P_i$ .  
     Let  $v_2, \dots, v_p$  be the second through penultimate vertices of  $P_i$ .  
     Insert  $v_2, \dots, v_p$  into  $\mathcal{L}$  after  $v_1$ .  
     For  $j = 2, \dots, p$ :  
         Assign the arc leaving  $v_j$  the interval  $(v_j, v_j)$ .  
     Let  $v_1u$  be the arc leaving  $v_1$  that is assigned an interval  $(v_1, a)$ .  
     Let  $b$  be the vertex after  $v_p$  in the cyclic ordering  $\mathcal{L}$ .  
     Reassign  $v_1u$  the interval  $[b, a)$ .  
     Assign the arc  $v_1v_2$  the interval  $(v_1, b)$ .

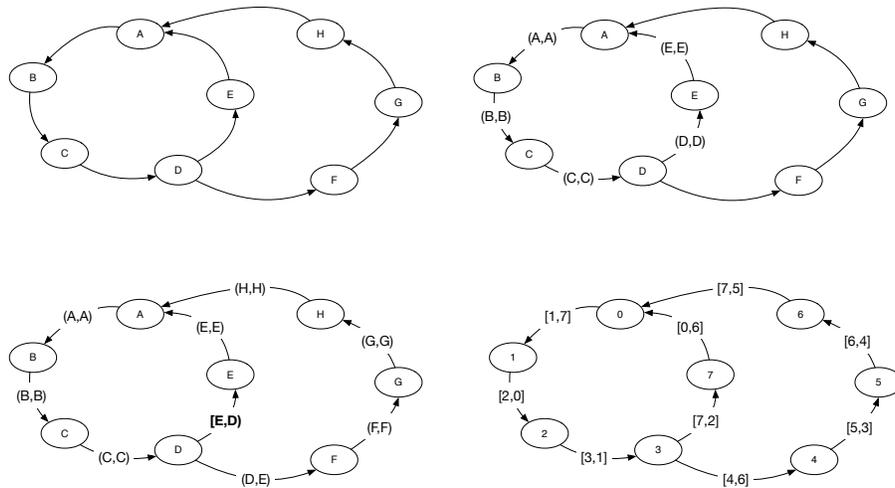


Figure 2: Top left: Input strongly connected component with symbolic vertex labels and two ears,  $P_0 = A, B, C, D, E$  and  $P_1 = D, F, G, H, A$ . Top right: Arc labeling of  $P_0$  with cyclic ordering  $\mathcal{L} = \{A, B, C, D, E\}$ ;  $(C, C)$  indicates the range of  $\mathcal{L}$  strictly after  $C$  and strictly before  $C$  (namely,  $D, E, A, B$ ). Bottom left: Arc labeling of  $P_1$ , update of the label for the arc(s)  $(DE)$  leaving the first vertex of  $P_1$  ( $D$ ), and inserting the internal vertices  $F, G, H$  of  $P_1$  into the cyclic ordering before the last vertex of  $P_1$ ;  $\mathcal{L} = \{A, B, C, D, F, G, H, E\}$ ;  $[E, D)$  indicates the range of  $\mathcal{L}$  after and including  $E$  and strictly before  $D$  (namely,  $E, A, B, C$ ). Bottom right: conversion to a numerical scheme with closed (cyclic) intervals by mapping the  $i^{th}$  element of  $\mathcal{L}$  to the number  $i$ .

The following invariant, among other things, shows that the arc  $v_1u$  exists and is unique.

**Invariant 4** *At any stage in the algorithm, the intervals assigned to the arcs leaving a vertex  $v$  are disjoint and form a partition of  $(v, v)$ .*

**Proof:** When a vertex is first introduced and there is only one arc leaving it, this invariant is true by construction.

If we assume for an induction that the invariant holds prior to the introduction of a new arc  $v_1v_2$  leaving  $v_1$ , then there must be exactly one arc whose assigned interval starts with  $(v_1, \cdot)$ . In fact, since a closed endpoint of an interval is never introduced, this arc must be assigned an interval of the form  $(v_1, a)$  for some  $a$ . Since, prior to the insertion of  $v_2, \dots, v_p$  into  $\mathcal{L}$ ,  $b$  is the vertex after  $v_1$  in the cycle ordering,  $(v_1, b) \subseteq (v_1, a)$  for all  $a$ . Therefore after the insertion of  $v_2, \dots, v_p$  into  $\mathcal{L}$  between  $v_1$  and  $b$  we still have that  $(v_1, b) \subseteq (v_1, a)$ . Since  $(v_1, b), [b, a)$  is a partition of  $(v_1, a)$ , the invariant holds.  $\square$

**Theorem 5** ROUTING produces a feasible interval routing scheme with each arc having exactly one interval.

**Proof:** As mentioned above, for convenience of analysis, we view the intervals as continuous. Let  $H_i = \cup_{j \leq i} P_j$  and let  $\mathcal{L}_i$  be the cyclic ordering of the vertices of  $H_i$  at the end of iteration  $i$ . For a vertex  $v \in H_i$ , let  $n_i(v)$  be the vertex immediately after  $v$  in  $\mathcal{L}_i$ . We prove the following statement by induction: for  $v \in H_i$ , at the end of iteration  $i$ , a message with destination in the (continuous) interval  $[v, n_i(v))$  will reach vertex  $v$ ; that is, at the end of iteration  $i$  the intervals provide a feasible routing scheme for  $H_i$ . This statement is true for the base case which corresponds to the interval assignment for  $P_0$ .

Consider the intervals defined at the end of iteration  $i$ . We show that these allow a message with destination in the interval  $[y, n_i(y))$  starting at vertex  $x$  will reach vertex  $y$  (for  $x \neq y$ ). The non-trivial cases are Cases 2, 3 and 4.

1.  $x$  and  $y$  are internal vertices of  $P_i$  and  $y$  is after  $x$  along  $P_i$ :  
 For every vertex  $v$  in  $P_i$ , a message will get routed on the arc leaving  $v$  unless it is destined for  $v$  since the interval assigned to the unique arc leaving  $v$  contains everything except  $v$ .
2.  $x = v_1$  and  $y$  is an internal vertex of  $P_i$ :  
 The internal vertices of  $P_i$  are in the interval  $[v_2, v_p]$ . By construction and definition of  $b$ , this is the same as the interval  $(v_1, b)$  since  $b$  is the vertex after  $v_p$  in  $\mathcal{L}_i$  and  $v_1$  is the vertex before  $v_2$  in  $\mathcal{L}_i$ . So  $[y, n_i(y)) \subset (v_1, b)$  and a message at  $v_1$  going to a destination in  $[y, n_i(y))$  gets routed along the arc  $v_1v_2$ . Correctness follows from Case 1.
3.  $x \in H_{i-1} \setminus v_1$  and  $y$  and internal vertex of  $P_i$ :  
 We argue that the message will reach  $v_1$ . By definition  $b = n_{i-1}(v_1)$  and by construction  $[y, n_i(y)) \subset [v_1, b) = [v_1, n_{i-1}(v_1))$ . Since  $v_1 \in H_{i-1}$ , by the inductive hypothesis, a message with destination in the interval  $[v_1, n_{i-1}(v_1))$  will reach  $v_1$ ; we are done by Case 2.
4.  $x = v_1$  and  $y \in H_{i-1}$ :  
 Note that  $n_{i-1}(y) = n_i(y)$ . Since  $y \neq x$  and  $b = n_{i-1}(x)$ ,  $y \in [b, v_1)$ . Therefore  $[y, n_i(y)) \cap (v_1, b)$  is empty and the message does not get routed along  $v_1v_2$ . Therefore a message in  $[y, n_i(y))$  reaches  $y$  by the inductive hypothesis.

5.  $x, y \in H_{i-1}$ :

If a message in  $[y, n_i(y))$  reaches  $v_1$ , then the message reaches  $y$  by Case 4. If a message in  $[y, n_i(y))$  does not reach  $v_1$ , then we are done by the inductive hypothesis because  $n_{i-1}(y) = n_i(y)$ .

6.  $x$  is an internal vertex of  $P_i$  and  $y \in H_{i-1}$ :

Note that  $n_{i-1}(y) = n_i(y)$ . Since  $v \notin [y, n_i(y))$  for any internal vertex  $v$  of  $P_i$ , a message in  $[y, n_i(y))$  will reach  $H_{i-1}$ . Then by Case 5, a message in  $[y, n_i(y))$  will reach  $y$ .

7.  $x$  and  $y$  are internal vertices of  $P_i$  and  $y$  is before  $x$  along  $P_i$ :

Since  $v \notin [y, n_i(y))$  for any internal vertex  $v$  of  $P_i$  after  $x$  because  $x$  is after  $y$  in  $P_i$ , a message in  $[y, n_i(y))$  will reach  $H_{i-1}$ . By Cases 2 and 3, a message in  $[y, n_i(y))$  will reach  $y$ .

□

It is non-standard to use open intervals for such a scheme. Given the final interval assignment and cyclic ordering, numbers can be assigned to the vertices based on the cyclic ordering and the intervals can be closed in the natural way.

### 3.2 Strongly-connected orientations that minimize the maximum indegree

We will show that a modified version of PATH-REVERSAL finds a strongly-connected orientation of an undirected graph that minimizes the maximum indegree. In this section we will assume that the given directed graph has a strongly-connected orientation. Given a directed graph, we say that a path from  $u$  to  $v$  is *strongly reversible* if  $\delta(u) < \delta(v) - 1$  and reversing the path will maintain strong connectivity. The greedy algorithm, given an undirected graph, is:

SC-PATH-REVERSAL

start with an arbitrary strongly-connected orientation

while there is a strongly reversible path starting with a max-indegree vertex

let  $P$  be such a path

reverse the orientation of each arc of  $P$

One can find a strongly-connected orientation in linear time using depth-first search: orient all edges in the depth-first search tree downward away from the root and orient all the non-tree edges upward with respect to the tree, cross edges may be oriented arbitrarily.

Strongly-reversible paths are characterized by the number of edge disjoint paths between endpoints. We say that a vertex  $v$  *two-reaches* a vertex  $u$  if there are two arc-disjoint paths from  $v$  to  $u$ . We say that a vertex  $v$  *two-reaches* a vertex set  $U$  if there are paths from  $v$  to  $u_1$  and from  $v$  to  $u_2$  where  $u_1, u_2 \in U$ , and these paths are arc disjoint.

In the following we will use network flow theory. Let  $N = (V, E)$  be a directed network with  $s, t \in V$  being the source and the sink of  $N$  respectively. The capacity of an edge is a mapping  $c : E \rightarrow \mathbb{R}^+$ , denoted by  $c_{uv}$ . This is the maximum amount of flow that can pass through an edge. A *flow* is a mapping  $f : E \rightarrow \mathbb{R}^+$ , denoted by  $f_{uv}$ , subject to the following two constraints:  $f_{uv} \leq c_{uv}$ , for each  $(u, v) \in E$  and  $\sum_{u:(u,v) \in E} f_{uv} = \sum_{u:(v,u) \in E} f_{vu}$ , for each  $v \in V \setminus \{s, t\}$ . The value of flow is defined by  $|f| = \sum_{v \in V} f_{sv}$ , where  $s$  is the source of  $N$ . An  $s - t$  cut  $C = (S, T)$  is a partition of  $V$  such that  $s \in S$  and  $t \in T$ . The cut-set of  $C$  is the set  $\{(u, v) \in E \mid u \in S, v \in T\}$ . The max-flow, min-cut theorem states that the value of the maximum flow is equal to the value of the minimum cut [8].

A consequence of this theorem is that if the maximum flow is greater than or equal to  $k$  in a unit-capacity network, then there are  $k$  arc disjoint  $s$  to  $t$  paths [1].

**Lemma 2** *Reversing a  $u$ -to- $v$  path maintains strong connectivity if and only if  $u$  two-reaches  $v$ .*

**Proof:** Suppose that when we reverse a  $u$ -to- $v$  path  $P$ , the graph remains strongly connected. Thus there must still be a  $u$ -to- $v$  path when  $P$  is reversed, so in the original graph the max  $u$ -to- $v$  flow must have been at least 2. By the max-flow, min-cut theorem, we know that there are 2 arc-disjoint paths from  $u$  to  $v$ .

Now suppose that  $u$  two-reaches  $v$  in a strongly connected orientation. Reversing one of these paths will create a cycle. Any pair of vertices requiring one of these paths for connectivity can be connected by way of the cycle, which will maintain strong connectivity.  $\square$

SC-PATH-REVERSAL can be implemented to run in quadratic time: strong-path reversibility can be detected in linear time by two iterations of the augmenting path algorithm for maximum flow [8]. There are a linear number of iterations: we show, as in Section 2, that after reversing a strongly-reversible path ending in a vertex of indegree  $\ell$ , the algorithm does not reverse a strongly-reversible path ending in a vertex of higher indegree.

As for the algorithm PATH-REVERSAL, we argue that the algorithm proceeds in  $k$  phases where  $k$  is the maximum indegree of the initial strongly connected orientation. In phase  $\ell$ , strongly-reversible paths ending in vertices of indegree  $\ell$  are reversed. This reduces the indegree of these vertices by one, and does not result in any extra vertices of indegree greater than  $\ell$ .

Let  $Q$  be the set of vertices of indegree  $> \ell$  just before the start of phase  $\ell$  and let  $Q'$  be the set of vertices that have strongly-reversible paths to a vertex in  $Q$ . (Note:  $Q \subseteq Q'$ .) If, in the first iteration of phase  $\ell$ , a strongly-reversible path ending in a vertex  $v$  of indegree  $\ell$  is reversed, then the indegrees of the vertices in  $Q'$  must be  $> \ell$  and  $v \notin Q$ . Therefore, after reversing a strongly-reversible path ending in a vertex of indegree  $\ell$ , the algorithm does not reverse a path ending in a vertex of higher indegree.

### 3.2.1 Strongly connected structure

To prove that SC-PATH-REVERSAL minimizes the maximum indegree, we will use a transitivity-like property of arc-disjointness:

**Lemma 3** *Suppose vertices  $s$  and  $t$  each two-reach a vertex  $v$ . If there are arc-disjoint  $u$ -to- $s$  and  $u$ -to- $t$  paths, then  $u$  two-reaches  $v$ .*

**Proof:** We argue that the min  $u$ - $v$  cut is at least 2, proving the lemma by the max-flow-min-cut theorem. Consider any  $u$ - $v$  cut (viewed as a bipartition of the vertices),  $(A, B)$ . If  $s \in A$ , then the min cut is at least 2 (because the min  $sv$ -cut is at least 2). Likewise if  $t \in A$ . If both  $s$  and  $t$  are in  $B$ , then the min  $u$ - $v$  cut is at least 2, as witnessed by the arc-disjoint  $u$ -to- $s$  and  $u$ -to- $t$  paths.  $\square$

In order to ensure strong connectivity, we get:

**Corollary 6** *Let  $U$  be a set of vertices. For each component  $C$  of  $G[U]$ , there must be at least one arc from each component of  $G[V \setminus C]$  to  $C$ .*

We can in fact meet the implied lower bound:

**Lemma 4** *Let  $v$  be a vertex of maximum indegree obtained by the SC-PATH-REVERSAL algorithm. Let  $U$  be the set of vertices that two-reach  $v$ . Each component of  $G[V \setminus U]$  has exactly one arc to  $U$  in the SC-PATH-REVERSAL orientation.*

**Proof:** Let  $C$  be a component of  $G[V \setminus U]$  and suppose for a contradiction that there are multiple arcs from  $C$  to  $U$ . Let  $v_1, v_2, v_3, \dots, v_p$  be vertices in  $C$  that are tails of these arcs. Let  $C_i$  be the set of vertices in  $C$  that reach  $v_i$ . We will argue that all of the  $C_i$ s are in fact the same, so there is only one  $v_i$  that is the tail of an arc from  $C$  into  $U$ .

Since the graph is strongly connected, every vertex in  $C$  reaches  $U$  and so must reach some  $v_i$ . If  $x \in C_i \cap C_j$  for some  $i \neq j$ , then by Lemma 3,  $x$  two-reaches  $U$ , contradicting the definition of  $U$ . Therefore  $C_1, \dots, C_p$  is a partition of  $C$ . However, since  $C$  is connected, there must be an arc  $uv$  from, say,  $C_i$  to  $C_j$ . In which case,  $u \in C_j$ , a contradiction by the above case. Therefore, the partition cannot contain more than one set. So there is only one arc from  $C$  to  $U$ .  $\square$

### 3.2.2 Minimizing the maximum indegree

We show that the algorithm minimizes the maximum indegree by meeting the following lower bound. For a set of vertices  $U$ , let  $c(U)$  be the number of components of  $G[V \setminus U]$ .

**Lemma 5** *The maximum indegree of any strongly connected orientation is at least*

$$\max_{U \subseteq V} \left\lceil \frac{m(U) + c(U)}{|U|} \right\rceil.$$

**Proof:** The total indegree that must be shared amongst  $U$  is at least the number of edges in  $G[U] + c(U)$ , where the second term follows from Corollary 6. By an averaging argument at least one vertex must have indegree at least  $\left\lceil \frac{m(U)+c(U)}{|U|} \right\rceil$ .  $\square$

**Theorem 7** *The algorithm SC-PATH-REVERSAL finds a strongly connected orientation that minimizes the maximum indegree.*

**Proof:** Let  $k$  be the maximum indegree resulting from SC-PATH-REVERSAL. Let  $v$  be a vertex of indegree  $k$ . Let  $U$  be the set of vertices that two-reach  $v$ . By the termination criteria of the algorithm, all vertices in  $U$  have indegree  $k$  or  $k - 1$ . By Lemma 4, the total indegree shared amongst  $U$  is  $m(U) + c(U)$ . We have that  $|U|k \geq m(U) + c(U) > |U|(k - 1)$ . Dividing by  $|U|$  yields  $k \geq \frac{m(U)+c(U)}{|U|} > (k - 1)$ . By Lemma 5, this is the best possible.  $\square$

Theorem 7 has previously been proven in a non-constructive manner by Frank [9, 10].

We conjecture that SC-PATH-REVERSAL is indeed optimal for the “minimizing the lexicographic order” objective as well. Unfortunately our proof technique from Section 2 for minimizing the lexicographic order of an arbitrary orientation does not follow through. For example, we would need to consider the set of vertices  $U$  that have at least two paths to a vertex of highest indegree, but there could be a vertex  $x$  on a path from  $u \in U$  to  $v$  that is not in  $U$ . In this case inequality (1) does not hold. For this, and other reasons, a different technique will be needed to obtain this result.

## 4 Acyclic orientations

We now examine the situation in which the resulting orientation needs to be acyclic. Unlike what we have seen, minimizing the lexicographic order is no longer polynomially solvable. However, a simple algorithm guarantees optimality of minimizing the maximum indegree.

### 4.1 Minimizing the maximum indegree

The following simple procedure minimizes the maximum indegree for an acyclic orientation.

**Stripping Procedure** Choose a vertex with minimum degree. Orient all incident edges into that vertex. Remove that vertex and its adjacent edges. Repeat.

**Theorem 8** *Stripping finds an acyclic orientation with maximum indegree minimized.*

**Proof:** Let  $k$  be the maximum indegree resulting from stripping and let  $v$  be a vertex with indegree  $k$ . Thus at some iteration,  $v$  had the minimum degree among the remaining vertices,  $U$ . Let  $H$  be the subgraph induced by  $U$  with

orientation inherited from an optimal orientation of the original graph.  $H$  must have a sink and since every vertex in  $H$  has degree at least  $k$ , this vertex must have indegree at least  $k$ .  $\square$

## 4.2 Acyclic minimum lexicographic orientation is NP-hard

We will show that the problem of minimizing the lexicographic order of an acyclic orientation is NP-hard. We will give a reduction from set cover to the related problem of finding an acyclic orientation such that:

- the maximum indegree is minimized
- the number of vertices with this maximum indegree is also minimized

We refer to the above problem as Min-Leading-Degree (MLD). Note that MLD is different from the problem of minimizing the lexicographic order. However, any optimal solution of the latter is an optimal solution of the former. Therefore, by proving MLD NP-hard, we can conclude that min lex is also NP-hard.

The *Set Cover* problem is defined as follows: Given a set of elements  $\{1, 2, \dots, m\}$  (called the universe) and  $n$  sets whose union comprises the universe, the set cover problem is to identify the smallest number of sets whose union contains all elements in the universe. Set cover is proven to be NP-hard by a reduction from the vertex cover problem [12].

We say that a set of vertices is *t-strippable* if the stripping procedure results in maximum indegree at most  $t$  among these vertices.

We will use the following *k-gadget graph*  $H_\ell$  with  $1 \leq \ell < k$  and  $k$  odd:

**If  $\ell$  is odd**  $H_\ell$  is composed of 2 copies of  $K_k$ , a clique on  $k$  vertices, and a root vertex  $r$ . Connect  $r$  to  $(k - \ell)/2$  of the vertices in each of the complete graphs. Add a matching between the vertices of degree  $k - 1$  of the  $K_k$  subgraphs. (Figure 3(a))

**If  $\ell$  is even**  $H_\ell$  is composed of a left and right copy of  $K_k$ , a root vertex  $r$ , and an extra vertex  $s$ . Connect  $r$  to  $k - \ell$  vertices in the left  $K_k$ . Connect  $s$  to  $\ell/2$  vertices in the left  $K_k$ . Connect  $s$  to  $k - \ell/2$  of the vertices in the right  $K_k$ . Add a matching between the vertices of degree  $k - 1$  of the left and right  $K_k$  subgraphs. (Figures 3(b) and 3(c))

It is easy to verify that  $H_\ell$  has the following properties:

1. All the vertices, except the root vertex  $r$ , have degree  $k$ .
2. The root vertex  $r$  has degree  $k - \ell$ .
3.  $H_\ell \setminus \{r\}$  is connected.
4.  $H_\ell$  is  $(k - 1)$ -strippable.

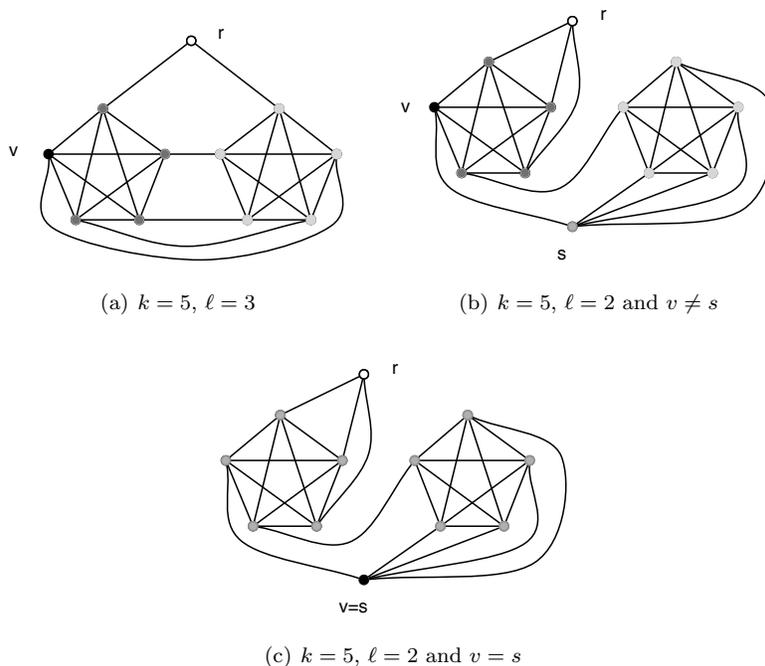


Figure 3: Above are examples for the gadgets when  $k = 5$  and in the case of Figure 3(a)  $\ell = 3$  or in Figures 3(b) and 3(c)  $\ell = 2$ . The shading denotes the stripping order as used in Lemma 6, we start by stripping the darkest vertex and lastly strip the lightest vertex.

Let  $\mathcal{S} = \{S_1, S_2, \dots, S_m\}$  be the instance of set cover. We wish to find  $\ell$  sets that cover all of the elements. Let  $f_x$  denote the frequency of element  $x$  in  $\mathcal{S}$ . Let  $k$  be the smallest odd number which is greater than  $\max_{i,x} \{|S_i|, f_x\}$ .

We construct a graph  $G$  as follows (See Figure 4):

For each  $S_i \in \mathcal{S}$  create a *set*  $k$ -gadget  $H_i$ , with root vertex  $r_i$ . For each element  $x$  create an *element*  $k$ -gadget  $H_{f_x}$ , with root vertex  $r_x$ . For every  $x \in S_i$  connect  $r_i$  to  $r_x$ .  $G$  has the following properties:

1. All vertices in  $G$  have degree  $k$  except for the vertices  $r_i$  of the set gadgets, these have degree  $k + |S_i| - 1$ .
2. All vertices have degree at least  $k$ , so the minimum possible maximum in-degree of any acyclic orientation of  $G$  is  $k$ .
3.  $G$  is  $k$ -strippable.

The first two properties are clear from construction. We will prove the third property with the following two lemmata.

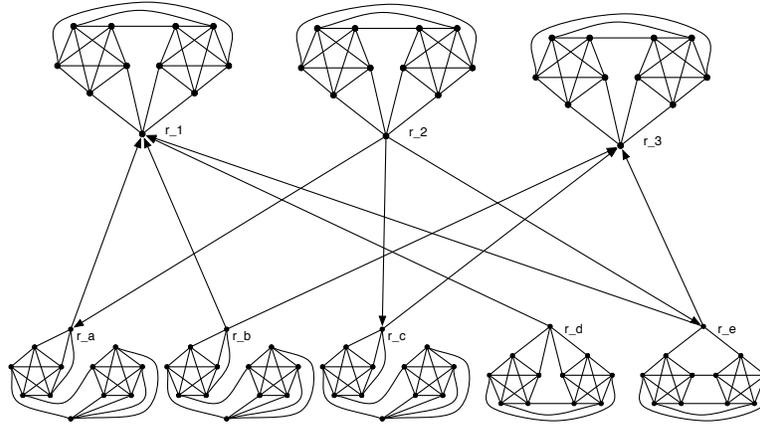


Figure 4: The following is an example of the graph corresponding to the Set Cover instance:  $\{a, b, c, d, e\}$  with sets  $\{S_1, S_2, S_3\}$ ,  $S_1 = \{a, b, d, e\}$ ,  $S_2 = \{a, c, e\}$ ,  $S_3 = \{b, c, e\}$ . Suppose that  $S_1$  and  $S_3$  are chosen for the cover, then we see all edges from the element gadgets are directed toward  $S_1$  and  $S_3$ .

**Lemma 6** *The vertices of a set gadget in  $G$  with any vertex  $v$  removed are  $(k - 1)$ -strippable.*

**Proof:**  $r$  is the only vertex in the set gadget that potentially has degree greater than  $k$ .

If  $v = r$  then the vertices that were adjacent to  $r$  have degree  $k - 1$  so we can  $(k - 1)$ -strip the remaining vertices.

Suppose that  $v \neq r$ . The vertices of the set gadget can be  $(k - 1)$ -stripped as illustrated in Figure 3.

If  $\ell$  is odd, first strip all of the vertices of the clique containing  $v$  then strip all of the vertices of the other clique. We know that  $r$  originally had degree  $k + |S_i| - 1$  and we removed  $k - 1$  of the vertices adjacent to it. Thus  $r$  has degree  $|S_i|$  which is at most  $k - 1$ , so we can strip  $r$ .

If  $\ell$  is even, then either  $v \in K_k$  or  $v = s$ . If  $v \in K_k$  for either the left or right clique, first strip all of the vertices in this clique, then strip  $s$ , then strip the vertices in the other clique and finally strip  $r$  as in the odd case. If  $v = s$  then we can strip both cliques, because both were connected to  $s$ , and finally strip  $r$ .  $\square$

A similar argument shows:

**Lemma 7** *Any element gadget with any vertex  $v$  removed is  $(k - 1)$ -strippable.*

It follows that  $G$  is  $k$ -strippable: stripping one vertex of degree  $k$  from each set and element gadget leaves  $(k - 1)$ -strippable subgraphs.

**Theorem 9**  *$G$  has an acyclic orientation with at most  $\ell$  vertices of indegree  $k$  if and only if there is a covering subcollection of  $\mathcal{S}$  of size at most  $\ell$ .*

**Proof:** For the forward direction: Let  $X$  be the set of at most  $\ell$  vertices which have indegree  $k$ .  $G \setminus X$  is  $(k - 1)$ -strippable. Let  $\mathcal{S}'$  be the subcollection of  $\mathcal{S}$  containing

- (i) all the sets whose gadgets have an indegree  $k$  vertex and
- (ii) for each element gadget that has an indegree  $k$  vertex, one set that contains this element.

Notice that  $|\mathcal{S}'| \leq \ell$ . We will show that  $\mathcal{S}'$  is a covering.

For any element  $x$ , let  $P$  be the element gadget corresponding to  $x$ . There are two cases for  $P$ :

- (a)  $P$  includes a vertex of indegree  $k$ .

In this case  $x$  is covered by a set of type (ii).

- (b)  $P$  does not include a vertex of indegree  $k$ .

The degree of the root vertex  $r_P$  of  $P$  is  $k$ . The assumption that  $P$  does not have any vertices of indegree  $k$  implies that at least one edge is oriented away from  $r_P$ . If this edge is from  $r_P$  to a set gadget, then at least one vertex of the set gadget must have indegree  $k$ . The set corresponding to this set gadget must be included in  $\mathcal{S}'$ , so  $x$  is covered. If the edge oriented away from  $r_P$  is directed to another vertex in  $P$ , then there must be a vertex of indegree  $k$  in  $P$ . By the construction of the gadget and the acyclicity property of the orientation, this is a contradiction to the fact that the edge oriented away from  $r_P$  is directed into another vertex in  $P$ . Thus in any case, element  $x$  is covered.

Therefore  $\mathcal{S}'$  is a cover.

For the reverse direction: Let  $\mathcal{S}'$  be the collection of at most  $\ell$  sets from  $\mathcal{S}$  that form a cover. Take a non-root vertex from each set gadget corresponding to a set in  $\mathcal{S}'$  and orient all edges toward it. Each of these gadgets is now  $(k - 1)$ -strippable by Lemma 6. Each element  $x$  is covered, so stripping the set gadget covering  $x$  directs the edge between the set gadget and the element gadget for  $x$  away from the element gadget (see the orientation of Figure 4). The root of the element gadget has degree  $k - 1$ , so each element gadget is  $(k - 1)$ -strippable in  $G \setminus X$ . Consider the set gadgets for the sets not in  $\mathcal{S}'$ . The roots of these gadgets all have remaining degree  $k - 1$ , because all of the element gadgets have been stripped so these are also  $k - 1$ -strippable. This orientation has at most  $\ell$  vertices of indegree  $k$ .  $\square$

**Corollary 10** *The problem of MLD is NP-hard, which implies the problem of minimizing the lexicographic order of indegrees is also NP-hard.*

## 5 Closing

Graph orientation is a rich problem area. In this paper we have presented three variants of the problem with their respective motivations. In one variant the resulting graph has no structural constraints, in another strong connectivity is required, and finally an acyclic orientation is required. For the first two variants

the simple algorithm of path reversal proves to be powerful. We have shown the optimality of the algorithm for the “minimizing the maximum indegree” objective in both variants and for the minimizing the lexicographic order objective for the first variant. We conjecture that SC-path-reversal is indeed optimal for the “minimizing the lexicographic order” objective as well. The third variant, requiring the resulting graph to be acyclic introduced quite a different problem. We have included an NP-hardness proof for acyclic minimizing the lexicographic order to demonstrate the point. How to approximate the “minimizing the lexicographic order” objective and enforce the acyclicity constraint remains an interesting open problem.

## Acknowledgements

We thank the reviewers for the insightful comments that helped us to clarify our work.

## References

- [1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.
- [2] Y. Asahiro, J. Jansson, E. Miyano, and H. Ono. Upper and lower degree bounded graph orientation with minimum penalty. In *Proc. Computing: The Australasian Theory Symposium*, pages 139–146, 2012.
- [3] Y. Asahiro, J. Jansson, E. Miyano, and H. Ono. Degree-constrained graph orientation: Maximum satisfaction and minimum violation. *Theor. Comp. Sys.*, 58(1):60–93, Jan. 2016. doi:10.1007/s00224-014-9565-5.
- [4] Y. Asahiro, J. Jansson, E. Miyano, H. Ono, and K. Zenmyo. Approximation algorithms for the graph orientation minimizing the maximum weighted outdegree. *J. Comb. Optim.*, 22(1):78–96, July 2011. doi:10.1007/s10878-009-9276-z.
- [5] Y. Asahiro, E. Miyano, H. Ono, and K. Zenmyo. Graph orientation algorithms to minimize the maximum outdegree. *International Journal of Foundations of Computer Science*, 18(2), 2007.
- [6] T. Biedl, T. Chan, Y. Ganjali, M. T. Hajiaghayi, and D. R. Wood. Balanced vertex-orderings of graphs. *Discrete Appl. Math.*, 148:27–48, April 2005. doi:10.1016/j.dam.2004.12.001.
- [7] S. Even and R. Tarjan. Computing an st-numbering. *Theoretical Computer Science*, 2:339–344, 1976.
- [8] C. Ford and D. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.
- [9] A. Frank. On the orientation of graphs. *Journal of Combinatorial Theory, Series B*, 28(3):251 – 261, 1980. doi:10.1016/0095-8956(80)90071-4.
- [10] A. Frank. *Connections in Combinatorial Optimization*. Oxford Lecture Series in Mathematics and Its Applications. OUP Oxford, 2011.
- [11] H. d. Fraysseix and P. O. de Mendez. Regular orientations, arboricity, and augmentation. In *Proceedings of the DIMACS International Workshop on Graph Drawing, GD '94*, pages 111–118, London, UK, UK, 1995. Springer-Verlag. URL: <http://dl.acm.org/citation.cfm?id=647546.730934>.
- [12] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.
- [13] C. Gavoille. A survey on interval routing. *Theoretical Computer Science*, 2000.

- [14] J. L. Gross and J. Yellen. *Handbook of Graph Theory*. CRC, Boca Raton, FL, USA, 2004.
- [15] W. F. Klostermeyer. Pushing vertices and orienting edges. *Ars Combinatorial*, 51:65–75, 1999.
- [16] J. V. Leeuwen and R. B. Tan. Interval routing. *The Computer Journal*, 4:298–307, 1987.
- [17] H. Robbins. A theorem on graphs with an application to a problem of traffic control. *Amer. Math. Monthly*, 46:281–283, 1939.
- [18] N. Santoro and R. Khatib. Labelling and implicity routing in networks. *The Computer Journal*, 28(1), 1985.
- [19] V. Venkateswaran. Minimizing maximum indegree. *Discrete Appl. Math.*, 143:374–378, September 2004. URL: <http://dx.doi.org/10.1016/j.dam.2003.07.007>, doi:10.1016/j.dam.2003.07.007.
- [20] H. Whitney. Non-separable and planar graphs. *Trans. Amer. Math. Soc.*, 34:339–362, 1932.
- [21] W. Wimmer. Ein Verfahren zur Verhinderung von Verklemmungen in Vermittlernetzen. <http://www.worldcat.org/title/verfahren-zur-verhinderung-von-verklemmungen-in-vermittlernetzen/>, October 1978.
- [22] V. Wittorff. Implementation of constraints to ensure deadlock avoidance in networks, 2009. US Patent # 7,532,584 B2.