Similarity-First Search: a new algorithm with application to Robinsonian matrix recognition

Monique Laurent[1,2] and Matteo Seminaroti[1]

[1]Centrum Wiskunde & Informatica (CWI), Science Park 123, 1098 XG Amsterdam, The Netherlands

[2]Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands

Abstract

We present a new efficient combinatorial algorithm for recognizing if a given symmetric matrix is Robinsonian, i.e., if its rows and columns can be simultaneously reordered so that entries are monotone nondecreasing in rows and columns when moving toward the diagonal. As main ingredient we introduce a new algorithm, named Similarity-First-Search (SFS), which extends Lexicographic Breadth-First Search (Lex-BFS) to weighted graphs and which we use in a multisweep algorithm to recognize Robinsonian matrices. Since Robinsonian binary matrices correspond to unit interval graphs, our algorithm can be seen as a generalization to weighted graphs of the 3-sweep Lex-BFS algorithm of Corneil for recognizing unit interval graphs. This new recognition algorithm is extremely simple and, for an $n \times n$ nonnegative matrix with m nonzero entries, it terminates in $n-1$ SFS sweeps, with overall running time $O(n^2 + nm \log n)$.

Keywords: *Robinson (dis)similarity; partition refinement; seriation; Lex-BFS; LBFS; Similarity Search*

1 Introduction

The seriation problem, introduced by Robinson [27] for chronological dating, is a classic and well known sequencing problem, where the goal is to order a given set of objects in such a way that similar objects are ordered close to each other (see e.g. [21] and references therein for details). This problem arises in many applications where objects are given through some information about their pairwise similarities (or dissimilarities) (like in data about user ratings, images, sounds, etc.).

The seriation problem can be formalized using a special class of matrices, namely Robinson matrices. A symmetric matrix $A = (A_{xy})_{x,y=1}^n$ is a *Robinson similarity matrix* if its entries are monotone nondecreasing in the rows and columns when moving toward the main diagonal, i.e., if $A_{xz} \le \min\{A_{xy}, A_{yz}\}$ for all $1 \le x < y < z \le n$. Given a set of n objects to order and a symmetric matrix $A = (A_{xy})$ whose entries represent their pairwise similarities, the seriation problem asks to find a permutation π of $[n]$ so that the matrix $A_\pi = (A_{\pi(x)\pi(y)})$, obtained by permuting both the rows and columns of A simultaneously

Correspondence to : M.Seminaroti@cwi.nl (M. Seminaroti), M.Laurent@cwi.nl (M. Laurent), CWI, Postbus 94079, 1090 GB, Amsterdam. Tel.:+31 (0)20 592 4386.

according to π, is a Robinson matrix. The matrix A is said to be a *Robinsonian similarity matrix* if such a permutation exists.

The Robinsonian structure is a strong property and, even though it might be desired in some problems, the data could be affected by noise, leading to the need to solve seriation in presence of error. Finding a Robinsonian matrix which is closest in the ℓ_∞-norm to a given similarity matrix is an NP-hard problem [6]. We refer to [7] for an approximation algorithm and to [17, 19] for approaches to this problem. Nevertheless, Robinsonian matrices play an important role also when data is affected by noise, as Robinsonian recognition algorithms can be used as core subroutines to design efficient heuristics or approximation algorithms for solving seriation in presence of errors (see, e.g., [7, 16]). In this paper we consider the problem of recognizing whether a given $n \times n$ matrix is Robinsonian.

In the past years, different recognition algorithms for Robinsonian matrices have been studied. The first polynomial algorithm to recognize Robinsonian matrices was introduced by Mirkin and Rodin [23]. It is based on the characterization of Robinsonian matrices in terms of interval hypergraphs, and it uses the PQ-tree algorithm of Booth and Leuker [3] as core subroutine, with an overall running time of $O(n^4)$. Chepoi and Fichet [5] introduced later a simpler algorithm using a divide-and-conquer strategy applied to preprocessed data obtained by sorting the entries of A, lowering the running time to $O(n^3)$. Using the same sorting preprocessing, Seston [29] improved the complexity of the recognition algorithm to $O(n^2 \log n)$. Recently, Préa and Fortin [25] presented an optimal $O(n^2)$ algorithm, using the algorithm from Booth and Leuker [3] to compute a first PQ-tree which they update throughout the algorithm. While all these algorithms use the connection to interval graphs or hypergraphs, in our previous work [20] we presented a recursive recognition algorithm exploiting a connection to unit interval graphs and with core subroutine Lexicographic Breadth-First Search (Lex-BFS or LBFS), a special version of Breadth-First Search (BFS) introduced by Rose and Tarjan [28]. The algorithm of [20] is suitable for sparse matrices and it runs in $O(d(m+n))$ time, where m is the number of nonzero entries of A and d is the depth of the recursion tree computed by the algorithm, which is upper bounded by the number of distinct nonzero entries of A.

While all the above mentioned recognition algorithms are combinatorial, Atkins et al. [1] presented earlier a numerical spectral algorithm, based on reordering the entries of the second smallest eigenvector of the Laplacian matrix associated to A (aka the Fiedler vector). Given its simplicity, this algorithm is used in some classification applications (see, e.g., [15]) as well as in spectral clustering (see, e.g., [2]), and it runs in $O(n(T(n)+n\log n))$ time, where $T(n)$ is the complexity of computing (approximately) the eigenvalues of an $n \times n$ symmetric matrix.

Note that the algorithms in [1], [25] and [20] also return all the possible Robinson orderings of a given Robinsonian matrix A, which can be useful in some practical applications.

In this paper we introduce a new combinatorial recognition algorithm for Robinsonian matrices. As a main ingredient, we define a new exploration algorithm for weighted graphs, named *Similarity-First Search* (SFS), which represents a generalization of the classical Lex-BFS algorithm to weighted graphs. Intuitively, the SFS algorithm explores vertices of a weighted graph in such a way that most similar vertices (i.e., corresponding to largest edge weights) are visited first, while still respecting the priorities imposed by previously visited vertices. When applied to an unweighted graph (or equivalently to a binary matrix), the SFS algorithm reduces to Lex-BFS. As for Lex-BFS, the SFS algorithm is entirely based on a unique simple task, namely partition refinement, a basic operation about sets which can be implemented efficiently (see [18] for details).

We will use the SFS algorithm to define our new Robinsonian recognition algorithm. Specifically, we introduce a multisweep algorithm, where each sweep uses the order returned by the previous sweep to break ties in the (weighted) graph search. Our main result in this paper is that our multisweep algorithm can recognize after at most $n-1$ sweeps whether a given $n \times n$ matrix A is Robinsonian. Namely we will show that the last sweep is a Robinson ordering of A if and only if the matrix A is Robinsonian. Assuming that the matrix A is nonnegative and given as an adjacency list of an undirected weighted graph with m nonzero entries, our algorithm runs in $O(n^2 + mn \log n)$ time.

Multisweep algorithms are well studied approaches to recognize classes of (unweighted) graphs. In the literature there exist many results on multisweep algorithms which are based on Lex-BFS. For example, interval graphs can be recognized in at most 5 sweeps [12], cographs and unit interval graphs in 3 sweeps (respectively, [4] and [8]). As a graph is a unit interval graph if and only if its adjacency matrix is Robinsonian [26], the 3-sweep recognition algorithm for unit interval graphs of Corneil [8] is in fact our main inspiration and motivation to develop a generalization of Lex-BFS for weighted graphs. Finally, Dusart and Habib [14] have recently introduced a multisweep algorithm to recognize in at most n sweeps cocomparability graphs. For a more exhaustive list of multisweep algorithms please refer to [9, 12].

Nevertheless, to the best of our knowledge, the present paper is the first work introducing and studying explicitely the properties of a multisweep search algorithm for weighted graphs. The only related idea that we could find is about replacing BFS with Dijkstra's algorithm, which is only briefly mentioned in [13].

The relevance of this work is twofold. First, we reduce the Robinsonian recognition problem to a single an extremely simple and basic operation, namely to partition refinement. Hence, even though from a theoretical point of view the algorithm is computationally slower than the optimal one presented in [25], its simplicity makes it easy to implement and thus hopefully will encourage the use and the study of Robinsonian matrices in more practical problems. Second, we introduce a new (weighted) graph search, which we believe is of independent interest and could potentially be used for the recognition of other structured matrices or just as basic operation in the broad field of 'Similarity Search'. In addition we introduce some concepts extending analogous notions in graphs and develop some combinatorial tools for the study of Robinsonian matrices that we need to analyze our new multisweep algorithm. As an example we give combinatorial characterizations for the end points (aka anchors) of Robinson orderings.

Contents of the paper

The paper is organized as follows. Section 2 contains some preliminaries. In Section 2.1 we give basic facts about Robinsonian matrices and Robinson orderings and we introduce several concepts (path avoiding a vertex, valid vertex, anchor) playing a crucial role in the paper. Section 2.2 contains combinatorial characterizations for (opposite) anchors of Robinsonian matrices.

Section 3 is devoted to the SFS algorithm. First, we describe the algorithm in Section 3.1 and we characterize SFS orderings in Section 3.2. Then, in Section 3.3 we introduce a fundamental lemma which we will use throughout the paper, named the 'Path

Avoiding Lemma'. Finally, in Section 3.4 we introduce the notion of 'good SFS ordering' and we show properties of end points of (good) SFS orderings, namely that they are (opposite) anchors of Robinsonian matrices.

In Section 4 we discuss the variant SFS$_+$ of the SFS algorithm, an extension of Lex-BFS$_+$ to weighted graphs, which differs from SFS in the way ties are broken The SFS$_+$ algorithm takes a given ordering as input which it uses to break ties. In Section 4.1 we show a basic property of the SFS$_+$ algorithm, namely that it 'flips' the end points of the input ordering. Then in Section 4.2 we introduce the 'similarity layers' of a matrix, a strengthened version of BFS layers for unweighted graphs, which are useful for the correctness proof of the multisweep algorithm. We show in particular that the similarity layers enjoy some compatibility with Robinson and SFS$_+$ orderings.

In Section 5 we present the multisweep algorithm to recognize Robinsonian matrices and we prove its correctness. In Section 5.1 we describe the multisweep algorithm and show that it terminates in 3 sweeps when applied to a binary matrix, thus giving a new proof of the result of Corneil [8] for unit interval graphs. In Section 5.2 we study properties of '3-good SFS orderings', which are orderings obtained after three SFS$_+$ sweeps. In particular we show that they contain classes of Robinson triples and that, after deleting their end points, they induce good SFS orderings, which will enable us to apply induction in the correctness proof. After that we have all the ingredients needed to conclude the correctness proof for the multisweep algorithm in Section 5.3.

Finally, in Section 6 we discuss the complexity of the SFS algorithm, and we conclude with remarks and open questions in Section 7.

2 Preliminaries

In this section we introduce some notation and recall some basic properties and definitions for unit interval graphs and Robinsonian matrices. In particular, we introduce the concepts of 'path avoiding a vertex' and 'valid vertex' and we give combinatorial characterizations for end points of Robinson orderings (also named 'anchors') and for 'opposite anchors', which will play an important role in the rest of the paper.

2.1 Basic facts

Let π be a linear order of $[n]$. For two distinct elements $x, y \in [n]$, the notation $x <_\pi y$ means that x appears before y in π and, for disjoint subsets $U, W \subseteq V$, $U <_\pi W$ means that $x <_\pi y$ for all $x \in U$, $y \in W$. The linear order π is a permutation of $[n]$, which can be represented as a sequence $(x_1, \dots, x_n)$ with $x_1 <_\pi \dots <_\pi x_n$, and π^{-1} is the reverse linear order $(x_n, x_{n-1}, \dots, x_1)$. An ordered partition $\phi = (B_1, \dots, B_r)$ of a ground set V is an ordered collection of disjoint subsets of V whose union is V.

Throughout, $\mathcal{S}^n$ denotes the set of symmetric $n \times n$ matrices. Given $A \in \mathcal{S}^n$ and a subset $S \subseteq [n]$, $A[S] = (A_{xy})_{x,y \in S}$ is the principal submatrix of A indexed by S. A symmetric matrix $A \in \mathcal{S}^n$ is called a *Robinson similarity matrix* if its entries are monotone nondecreasing in the rows and columns when moving towards the main diagonal, i.e., if

$$A_{xz} \leq \min\{A_{xy}, A_{yz}\} \quad \text{for all} \quad 1 \leq x < y < z \leq n. \tag{1}$$

Note that the diagonal entries of A do not play a role in the above definition. If there exists a permutation π of $[n]$ such that the matrix $A_\pi := (A_{\pi(x)\pi(y)})_{x,y=1}^n$, obtained by permuting both the rows and columns of A simultaneously according to π, is a Robinson matrix then A is said to be a *Robinsonian similarity* and π is called a *Robinson ordering* of A. In the literature, a distinction is made between Robinson(ian) similarities and Robinson(ian) dissimilarities. A symmetric matrix A is called a *Robinson dissimilarity matrix* if its entries are monotone nondecreasing in the rows and columns when moving away from the main diagonal. Hence $A \in \mathcal{S}^n$ is a Robinson(ian) similarity precisely when $-A$ is a Robinson(ian) dissimilarity and thus the properties extend directly from one class to the other one. For this reason, in this paper we will deal exclusively with Robinson(ian) similarities. Hence, when speaking of a Robinson(ian) matrix, we mean a Robinson(ian) similarity matrix. Furthermore, with $J \in \mathcal{S}^n$ denoting the all-ones matrix, it is clear that if A is a Robinson(ian) matrix then $A + \lambda J$ is also a Robinson(ian) matrix for any scalar λ. Hence, we may consider, without loss of generality, nonnegative similarities A (whose smallest entry is equal to 0).

In order to fully understand Robinsonian matrices and the motivation for our work, it is useful to briefly discuss the special class of binary Robinsonian matrices. Any symmetric matrix $A \in \{0,1\}^{n\times n}$ corresponds to a graph $G = (V = [n], E)$ whose edges are the positions of the nonzero entries of A. Then it is well known that A is a Robinsonian similarity if and only if G is a unit interval graph [26]. A graph $G = (V = [n], E)$ is called a *unit interval graph* if its vertices can be mapped to unit intervals $I_1, \ldots, I_n$ of the real line such that two distinct vertices $x, y \in V$ are adjacent in G if and only if $I_x \cap I_y \neq \emptyset$. There exist several equivalent characterizations for unit interval graphs. The following one highlights the analogy between unit interval graphs and Robinson orderings.

Theorem 2.1 (**3-vertex condition**). *[22] A graph $G = (V, E)$ is a unit interval graph if and only if there exists a linear ordering π of V such that, for all $x, y, z \in V$,*

$$x <_\pi y <_\pi z,\ \{x,z\} \in E \Longrightarrow \{x,y\}, \{y,z\} \in E. \tag{2}$$

It is clear that, for a binary matrix $A \in \mathcal{S}^n$, condition (1) is equivalent to (2). This equivalence and the fact that unit interval graphs can be recognized with a Lex-BFS multisweep algorithm [8] motivated us to find an extension of Lex-BFS to weighted graphs and to use it to obtain a (simple) multisweep recognition algorithm for Robinsonian matrices.

Given the analogy with unit interval graphs, it will be convenient to view symmetric matrices as weighted graphs. Namely, any nonnegative symmetric matrix $A \in \mathcal{S}^n$ corresponds to the weighted graph $G = (V = [n], E)$ whose edges are the pairs $\{x, y\}$ with $A_{xy} > 0$, with edge weights A_{xy}. Again, the assumption of nonnegativity can be made without loss of generality and is for convenience only. Accordingly we will often refer to the elements of $V = [n]$ indexing A as vertices (or nodes). For $x \in V$, $N(x) = \{y \in V \setminus \{x\} : A_{xy} > 0\}$ denotes the neighborhood of x in G.

In what follows we will extend some graph concepts to the general setting of weighted graphs (Robinsonian matrices). Throughout the paper, we will point out links between our results and some corresponding known results for Lex-BFS applied to graphs and we will mostly refer to [12] where more complete references about Lex-BFS can be found.

We now introduce some notions and simple facts about Robinsonian matrices and orderings. Consider a matrix $A \in \mathcal{S}^n$. Given distinct elements $x, y, z \in V$, the triple (x, y, z) is said to be *Robinson* if it satisfies (1), i.e., if $A_{xz} \le \min\{A_{xy}, A_{yz}\}$. Given a set $S \subseteq V$ and $x \in V \setminus S$, we say that x is *homogeneous* with respect to S if $A_{xy} = A_{xz}$ for all $y, z \in S$ (extending the corresponding notion for graphs, see, e.g., [12]). The following is an easy necessary condition for the Robinson property.

Lemma 2.2. *Let $A \in \mathcal{S}^n$ be a Robinsonian similarity. Assume that there exists a Robinson ordering π such that $x <_\pi z <_\pi y$. Then $A_{uz} \geq \min\{A_{ux}, A_{uy}\}$ for all $u \neq x, y, z \in [n]$.*

Proof. Indeed, $u <_\pi z$ implies $u <_\pi z <_\pi y$ and thus $A_{uz} \geq A_{uy}$, and $z <_\pi u$ implies $x <_\pi z <_\pi u$ and thus $A_{uz} \geq A_{ux}$. □

We now make a simple observation on how three elements $x, y, z \in V$ may appear in a Robinson ordering π of A depending on their similarities. Namely, if we have that $A_{xz} > \min\{A_{xy}, A_{yz}\}$ then, either y comes before both x and z in π, or y comes after both x and z in π. In other words, if x and z are more similar to each other than to y, then y cannot be ordered between x and z in any Robinson ordering π. Moreover, if $A_{xz} < \min\{A_{xy}, A_{yz}\}$ then, either $x <_\pi y <_\pi z$, or $z <_\pi y <_\pi x$. In other words, if x and z are more similar to y than to each other, then y must be ordered between x and z in any Robinson ordering π.

This observation motivates the following notion of 'path avoiding a vertex', which will play a central role in our discussion. Note that this notion is closely related to the notion of 'path missing a vertex' for Lex-BFS [12], although it is not equivalent to it when applied to a binary matrix.

Definition 2.3 (Path avoiding a vertex). *Given distinct elements $x, y, z \in V$, a* path from x to z avoiding y *is a sequence $(x = v_0, v_1, \ldots, v_{k-1}, v_k = z)$ of elements of V where each triple (v_i, y, v_{i+1}) is not Robinson, i.e.,*

$$A_{v_i v_{i+1}} > \min\{A_{yv_i}, A_{yv_{i+1}}\}, \quad \forall\ i = 0, 1, \ldots, k-1.$$

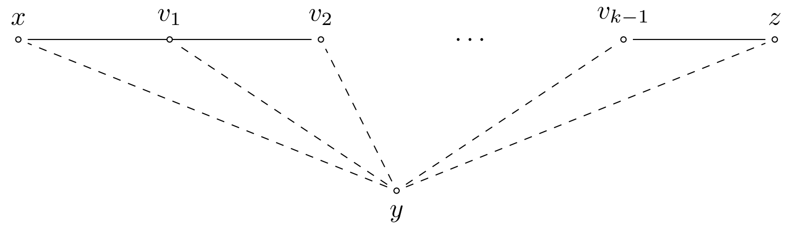

Figure 1: A path from x to z avoiding y: each continuous line indicates a value which is strictly larger than the minimum of the two adjacent dotted lines

The following simple but useful property holds.

Lemma 2.4. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix. If there exists a path from x to z avoiding y, then y cannot lie between x and z in any Robinson ordering π of A.*

Proof. Let $(x = v_0, v_1, \ldots, v_{k-1}, v_k = z)$ be a path from x to z avoiding y. Then, by definition, we have $A_{v_i v_{i+1}} > \min\{A_{yv_i}, A_{yv_{i+1}}\}$ for all $i = 0, 1, \ldots, k-1$, and thus y cannot appear between v_i and v_{i+1} in any Robinson ordering π. Hence y cannot lie between x and z in any Robinson ordering π. □

We now introduce the notion of 'valid vertex' which we will use in the next section to characterize end points of Robinson orderings.

Definition 2.5 (Valid vertex). *Given a matrix $A \in \mathcal{S}^n$, an element $z \in V$ is said to be* valid *if, for any distinct elements $u, v \in V \setminus \{z\}$, there do not exist both a path from u to z avoiding v and a path from v to z avoiding u.*

Observe that, if $z \in V$ is a valid vertex of a matrix A and $S \subseteq V$ is a subset containing z, then z is also a valid vertex of $A[S]$. It is easy to see that, for a $0-1$ matrix, the above definition of valid vertex coincides with the notion of valid vertex for Lex-BFS [12].

Consider, for example, the following matrix (already ordered in a Robinson form):

$$A = \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \\ f \\ g \end{array} \begin{array}{c} \begin{array}{ccccccc} a & b & c & d & e & f & g \end{array} \\ \begin{pmatrix} * & 7 & 6 & 0 & 0 & 0 & 0 \\ & * & 7 & 3 & 2 & 1 & 1 \\ & & * & 7 & 2 & 2 & 1 \\ & & & * & 3 & 3 & 3 \\ & & & & * & 7 & 5 \\ & & & & & * & 6 \\ & & & & & & * \end{pmatrix} \end{array}$$

Then the vertex d is not valid. Indeed, for the two vertices a and g, there exist a path from a to d avoiding g and a path from g to d avoiding a; namely the path (d, b, a) avoids g and the path (d, b, g) avoids a (see Figure 2).

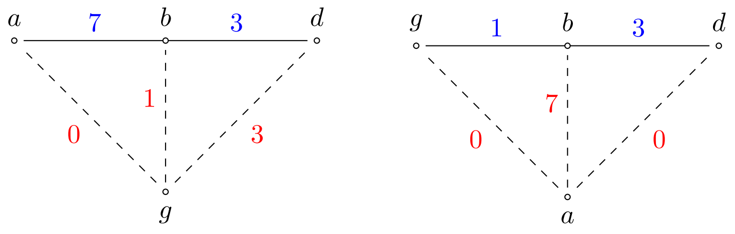

Figure 2: Element d is not admissible

2.2 Characterization of anchors

In this section we introduce the notion of '(opposite) anchors' of a Robinsonian matrix and then we give characterizations in terms of valid vertices. The notion of anchor was used for unit interval graphs in [10] and it is the analogue of the notion of end-vertex for interval graphs [12].

Definition 2.6 (**Anchor**). *Given a Robinsonian similarity $A \in \mathcal{S}^n$, a vertex $a \in [n]$ is called an* anchor *of A if there exists a Robinson ordering π of A whose last vertex is a. Moreover, two distinct vertices a, b are called* opposite anchors *of A if there exists a Robinson ordering π of A with a as first vertex and b as last vertex.*

Hence, an anchor is an end point of a Robinson ordering. Clearly, every Robinsonian matrix has at least one pair of opposite anchors. It is not difficult to see that every anchor must be valid. We now show that conversely every valid vertex is an anchor. This is the analogue of [9, Lemma 2] for Lex-BFS over interval graphs.

Theorem 2.7. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix. Then a vertex $z \in V$ is an anchor of A if and only if it is valid.*

Proof. ($\Rightarrow$) Assume z is an anchor of A and let π be a Robinson ordering of A with z as last element. Suppose for contradiction that, for some elements $u, v \in V$, there exist both a path P from u to z avoiding v and a path Q from v to z avoiding u. Using Lemma 2.4 and the path P, we obtain that that v lies before u or after z in π, and using the path Q we obtain that u lies before v or after z in π. As z is the last element of π, we must have $v <_\pi u$ in the first case and $u <_\pi v$ in the second case, which is impossible.

($\Leftarrow$) Conversely, assume that z is valid; we show that z is an anchor of A. The proof is by induction on the size n of the matrix A. The result holds clearly when $n = 2$. So we now assume $n \geq 3$ and that the result holds for any Robinsonian matrix of order at most $n - 1$. We need to construct a Robinson ordering π' of A with z as last vertex. For this we consider a Robinson ordering π of A. We let x denote its first element and y denote its last element. If $z = x$ or $z = y$, then we would be done. Hence we may assume $x <_\pi z <_\pi y$. For any $v <_\pi z$, we denote by $P_\pi(v, z)$ the path from v to z consisting of the sequence of vertices appearing consecutively between v and z in π.

We now define the following two sets:

$$\mathcal{B} = \{v <_\pi z : P_\pi(v, z) \text{ avoids } y\}, \quad \mathcal{C} = \{v <_\pi z : v \notin \mathcal{B}\}. \tag{3}$$

Next we show their following properties, which will be useful to conclude the proof.

Claim 1. *The following holds:*

(i) For any $v \in \mathcal{B}$, $A_{vy} = A_{yz}$.

(ii) If $v \in \mathcal{B}$ and $v <_\pi u <_\pi z$, then $u \in \mathcal{B}$.

(iii) Any element $v \in \mathcal{C}$ is homogeneous with respect to $V \setminus \mathcal{C}$, i.e., $A_{vw} = A_{vw'}$ for all $w, w' \in V \setminus \mathcal{C}$.

Proof. (i) As $v <_\pi z <_\pi y$, then $A_{vy} \leq A_{yz}$. We show that equality holds. Suppose not, i.e., $A_{vy} < A_{yz}$. Then $Q = (y, z)$ is a path from y to z avoiding v. Since $v \in \mathcal{B}$, $P = P_\pi(v, z)$ is a path from v to z avoiding y, and thus the existence of the paths P, Q contradicts the assumption that z is valid. Hence we must have $A_{vy} = A_{yz}$.

(ii) If $v \in \mathcal{B}$ then $P_\pi(v, z)$ avoids y and thus the subpath $P_\pi(u, z)$ also avoids y, which implies $u \in \mathcal{B}$.

(iii) Let $u \in \mathcal{B}$ denote the element of $\mathcal{B}$ appearing first in the Robinson ordering π. Then, for any $v \in \mathcal{C}$, $v <_\pi u <_\pi y$ and thus $A_{vy} \leq A_{vu}$ by definition of Robinson ordering. Hence, in order to show that v is homogeneous with respect to $V \setminus \mathcal{C}$, it suffices to show that $A_{vu} = A_{vy}$ (as, using the Robinson ordering property, this would in turn imply that $A_{vw} = A_{vw'}$ for all $w, w' \in V \setminus \mathcal{C}$). Suppose for contradiction that there exists $v \in \mathcal{C}$ such that $A_{vu} \neq A_{vy}$, and let v denote the element of $\mathcal{C}$ appearing last in π with $A_{vu} \neq A_{vy}$.

Then $A_{vu} > A_{vy}$ and the path (v, u) avoids y. Since $P_\pi(u, z)$ is a path from u to z avoiding y (because $u \in \mathcal{B}$), then the path $P = \{v\} \cup P_\pi(u, z)$ (obtained by concatenating (v, u) and $P_\pi(u, z)$) is a path from v to z avoiding y. This implies that v and u cannot be consecutive in π, as otherwise we would have $v \in \mathcal{B}$, contradicting the fact that $v \in \mathcal{C}$. Hence, there exists $v' \in \mathcal{C}$ such that $v <_\pi v' <_\pi u$. By the maximality assumption on v, it follows that $A_{v'u} = A_{v'y}$.

As z is valid and $P = \{v\} \cup P_\pi(u, z)$ is a path from v to z avoiding y, it follows that no path from y to z can avoid v. In particular, the path (y, z) does not avoid v and thus it must be $A_{yz} \leq \min\{A_{vy}, A_{vz}\}$. Recall that we assumed $A_{vu} > A_{vy}$. As $v <_\pi v' <_\pi u <_\pi z <_\pi y$, combining the above inequalities with the inequalities coming from the Robinson ordering π, we obtain $A_{v'y} \leq A_{yz} \leq A_{vy} < A_{vu} \leq A_{v'u}$, which contradicts the equality $A_{v'u} = A_{v'y}$. □

We now turn to the set of vertices coming after z in π. Symmetrically with respect to z, we can define the analogues of the sets $\mathcal{C}, \mathcal{B}$ defined in (3), which we denote by $\mathcal{C}', \mathcal{B}'$. For this replace π by its reverse ordering π^{-1} and y by x (the first element of π and thus the last element of π^{-1}), i.e., set

$$\mathcal{B}' = \{v >_\pi z : P_\pi(z, v) \text{ avoids } x\}, \quad \mathcal{C}' = \{v >_\pi z : v \notin \mathcal{B}'\}.$$

To recap, we have that $\pi = (\mathcal{C}, \mathcal{B}, z, \mathcal{B}', \mathcal{C}')$. Recall that x and y are respectively the first and the last vertex in π. Note that it cannot be that $\mathcal{C} = \mathcal{C}' = \emptyset$, as this would imply that $x \in \mathcal{B}$ and $y \in \mathcal{B}'$, and thus this would contradict the fact that z is valid (using the definition of the two sets $\mathcal{B}$ and $\mathcal{B}'$). Therefore, we may assume (without loss of generality) that $\mathcal{C} \neq \emptyset$. Let v be the vertex of $\mathcal{C}$ appearing last in the Robinson ordering π. By Claim 1 (iii), v is homogeneous with respect to the set $S = V \setminus \mathcal{C}$, i.e., all entries A_{vw} take the same value for any $w \in S$.

Consider the matrix $A[S]$, the principal submatrix of A with rows and columns in S. As $|S| \leq n - 1$ and z is valid (also with respect to $A[S]$), we can conclude using the induction assumption that z is an anchor of $A[S]$. Hence, there exists a Robinson ordering σ of $A[S]$ admitting z as last element.

Now, consider the linear order $\pi' = (\pi[\mathcal{C}], \sigma)$ of V obtained by concatenating first the order π restricted to $\mathcal{C} = V \setminus S$ and second the linear order σ of S. Using the fact that every vertex in $\mathcal{C}$ is homogeneous to all elements of S, we can conclude that the new linear order π' is a Robinson ordering of the matrix A. As z is the last element of π', this shows that z is an anchor of A and thus concludes the proof. □

The above proof can be extended to characterize pairs of opposite anchors.

Theorem 2.8. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix. Two distinct vertices $z_1, z_2 \in [n]$ are opposite anchors of A if and only if they are both valid and there does not exist a path from z_1 to z_2 avoiding any other vertex.*

Proof. ($\Rightarrow$) Assume that z_1 and z_2 are opposite anchors. Then they are both anchors and thus, in view of Theorem 2.7, they are both valid. Let π a Robinson ordering starting with z_1 and ending with z_2. Suppose, for the sake of contradiction, that there exists a vertex x and a path from z_1 to z_2 avoiding x. Then, by Lemma 2.4, x cannot lie in π between z_1 and z_2, yielding a contradiction.

($\Leftarrow$) Assume that z_1 and z_2 are valid and that there does not exist a path from z_1 to z_2 avoiding any other vertex. We show that they are opposite anchors. Consider a Robinson ordering π of A whose first element is z_1 and call y its last element. If $y = z_2$ then we are done. Hence, we may assume that $z_1 <_\pi z_2 <_\pi y$. As in the proof of Theorem 2.7, for any $v <_\pi z_2$, we denote by $P_\pi(v, z_2)$ the path from v to z_2 consisting of the sequence of vertices appearing consecutively between v and z_2 in π. Then, we can define the sets as in (3) in the proof of Theorem 2.7, where z is replaced by z_2, i.e.,:

$$\mathcal{B} = \{v <_\pi z_2 : P_\pi(v, z_2) \text{ avoids } y\}, \quad \mathcal{C} = \{v <_\pi z_2 : v \notin \mathcal{B}\}.$$

By assumption, $z_1 \notin \mathcal{B}$, else $P_\pi(z_1, z_2)$ would avoid y, contradicting the nonexistence of a path from z_1 to z_2 avoiding any other vertex. Therefore $z_1 \in \mathcal{C}$ and thus $\mathcal{C} \neq \emptyset$. Let $S = V \setminus \mathcal{C}$. Using the same reasoning as in the proof of Theorem 2.7, we can now conclude that one can find a Robinson ordering σ of $A[S]$, where S contains all the elements coming after the last element of $\mathcal{C}$ in π. The new linear order $\pi' = (\pi[\mathcal{C}], \sigma)$ of V obtained by concatenating first the order π restricted to $\mathcal{C} = V \setminus S$ and second the linear order σ of S is then a Robinson ordering of A whose first element is z_1 and whose last element is z_2, which concludes the proof. □

3 The SFS algorithm

In this section we introduce our new Similarity-First Search (SFS) algorithm. This algorithm will be applied to a (nonnegative) matrix $A \in \mathcal{S}^n$ and return a linear order of $V = [n]$, called a *SFS ordering* of A. As mentioned above, one can associate to A a weighted graph $G = (V = [n], E)$, with edges the pairs $\{x, y\}$ such that $A_{xy} > 0$ and edge weights A_{xy}. The SFS algorithm can be thus seen as a search algorithm for weighted graphs.

We first describe the algorithm in detail in Section 3.1 and provide a 3-point characterization of SFS orderings in Section 3.2. Then in Section 3.3 we discuss some properties of SFS orderings of Robinsonian matrices. Specifically, we introduce the fundamental 'Path Avoiding Lemma' (Lemma 3.6) which will be used repeatedly throughout the paper. In particular we use it in Section 3.4 to show a fundamental property of SFS orderings, namely that the last element of the SFS ordering of a Robinsonian matrix A is an anchor of A.

3.1 Description of the SFS algorithm

The SFS algorithm is a generalization of Lex-BFS for weighted graphs. As we will remark later, when applied to a $0-1$ matrix, the SFS algorithm coincides with Lex-BFS. Roughly speaking, the basic idea is to explore a weighted graph by visiting first vertices which are similar to each other (i.e., corresponding to an edge with largest weight) but respecting the priorities imposed by previously visited vertices. The algorithm is based on the implementation of Lex-BFS as a sequence of partition refinement steps as in [18].

Partition refinement is a simple technique introduced in [24] to refine a given ordered partition $\phi = (B_1, \dots, B_r)$ of the ground set V by a subset $W \subseteq V$. It produces a new ordered partition of V obtained by splitting each class B_i of ϕ in two sets, the intersection $B_i \cap W$ and the difference $B_i \setminus W$. If one visualizes an ordered partition as a priority list, the idea behind partition refinement is to modify the classes of the ordered partition while respecting the priorities among the vertices.

In our new SFS algorithm, we basically operate a sequence of partition refinements. But instead of splitting into two subsets we will split into several subsets. Specifically, given two ordered partitions ϕ and ψ, the output will be a new ordered partition which, roughly speaking, is obtained by splitting each class of ϕ into its intersections with the classes of ψ. The formal definition is as follows.

Definition 3.1 (Refine). *Let $\phi = (B_1, \dots, B_r)$ and $\psi = (C_1, \dots, C_s)$ be two ordered partitions of a set V and a subset $W \subseteq V$, respectively.* Refining ϕ by ψ *creates the new ordered partition of V, denoted by Refine(ϕ, ψ), obtained by replacing in ϕ each class B_i by the ordered sequence of classes $(B_i \cap C_1, \dots, B_i \cap C_s, B_i \setminus (C_1 \cup \dots \cup C_s) = B_i \setminus W)$ and keeping only nonempty classes.*

We will use this partition refinement operation in the case when the partition ψ is obtained by partitioning for decreasing values the elements of the neighborhood $N(p)$ of a given element p, according to the following definition.

Definition 3.2 (**Similarity partition**). *Consider a nonnegative matrix $A \in \mathcal{S}^n$ and an element $p \in [n]$. Let $a_1 > \ldots > a_s > 0$ be the distinct values taken by the entries A_{px} of A for $x \in N(p) = \{y \in [n] : A_{py} > 0\}$ and, for $i \in [s]$, set $C_i = \{x \in N(p) : A_{px} = a_i\}$. Then we define $\psi_p = (C_1, \ldots, C_s)$, which we call the similarity partition of $N(p)$ with respect to p.*

We can now describe the SFS algorithm. The input is a nonnegative matrix $A \in \mathcal{S}^n$ and the output is an ordering σ of the set $V = [n]$, that we call a *SFS ordering* of A. As in any general graph search algorithm, the central idea of the SFS algorithm is that, at each iteration, a special vertex (called the *pivot*) is chosen among the subset of unvisited vertices (i.e., the subset of vertices that have not been a pivot in prior iterations). Such vertices are ordered in a queue which defines the priorities for visiting them. Intuitively, the pivot is chosen as the most similar to the visited vertices, but respecting the visiting priorities imposed by previously visited vertices.

Algorithm 1: *SFS*(A)

```
input: a nonnegative matrix $A \in \mathcal{S}^n$
output: a linear order $\sigma$ of $[n]$

1  $\phi = (V) \leftarrow$ queue of unvisited vertices
2  for $i = 1, \ldots, n$ do
3      $S$ is the first class of $\phi$
4      choose $p$ arbitrarily in $S \leftarrow$ new pivot
5      $\sigma(p) = i \leftarrow$ let $p$ appear at position $i$ in $\sigma$
6      remove $p$ from $\phi$
7      $N(p)$ is the set of vertices $y \in \phi$ with $A_{py} > 0$
8      $\psi_p$ is the similarity partition of $N(p)$ with respect to $p$
9      $\phi =$ Refine $(\phi, \psi_p)$
10 return: $\sigma$
```

We now discuss in detail how the algorithm works. In the beginning, all vertices in V are unvisited, i.e., the queue ϕ of unvisited vertices is initialized with the unique class V.

At the iteration i, we are given an element p_{i-1} (which is the pivot chosen at iteration $i-1$) and a queue $\phi(p_{i-1}) = (B_1, \ldots, B_r)$, which is an ordered partition of the set of unvisited vertices. There are two main tasks to perform: the first task is to select the new pivot p_i, and the second task is to update the queue $\phi(p_{i-1})$ in order to obtain the new queue $\phi(p_i)$.

The first task is carried out as follows. As in the standard Lex-BFS, we denote by S the *slice* induced by p_{i-1} (i.e., the last visited vertex), which consists of the vertices among which to choose the next pivot p_i. The slice S coincides exactly with the first class B_1 of $\phi(p_{i-1})$. We distinguish two cases depending on the size of the slice S. If $|S| = 1$, then the new pivot p_i is the unique element of the slice S. If $|S| > 1$, we say that we have *ties* and, in the general version of the SFS algorithm, we break them arbitrarily. We will see in Section 4 a variant of SFS (denoted by SFS$_+$) where such ties are broken using a linear order given as additional input to the algorithm. Once the new pivot p_i is chosen, we mark it as visited (i.e., we remove it from the queue $\phi(p_{i-1})$) and we set $\sigma(p_i) = i$ (i.e., we let p_i appear at position i in σ).

The second task is the update of the queue $\phi(p_{i-1})$, which can be done as follows. Intuitively, we update $\phi(p_{i-1})$ according to the similarities of p_i with respect to the unvisited vertices and compatibly with the queue order. Specifically, first we compute the similarity partition $\psi_{p_i} = (C_1, \ldots, C_s)$ of the neighborhood $N(p_i)$ of p_i among the unvisited vertices (see Definition 3.2). Second, we refine the ordered partition $\phi(p_{i-1}) \setminus p_i = (B_1 \setminus \{p_i\}, B_2, \ldots, B_r)$ by the ordered partition ψ_{p_i} (see Definition 3.1). The resulting ordered partition is the ordered partition $\phi(p_i)$.

Note that if the matrix has only $0-1$ entries then the similarity partition ψ_{p_i} has only one class, equal to the neighborhood of p_i among the unvisited vertices. Hence, the refinement procedure defined in Definition 3.1 simply reduces to the partition refinement operation defined in [18] for Lex-BFS. This is why Lex-BFS is actually a special case of SFS for $0-1$ matrices.

Note also that, by construction, each class of the queue $\phi(p_i)$ is an interval of σ (i.e., the elements of the class are consecutive in σ). Furthermore, each of the visited vertices $p_1, \ldots, p_i$ is homogeneous to every class of the queue $\phi(p_i)$.

We show a simple example to illustrate how the algorithm works concretely. Consider the following matrix:

$$A = \begin{array}{c|cccccccccccc}
 & 1 & 2 & 3 & 5 & 7 & 8 & 9 & 11 & 13 & 14 & 17 & 19 \\
\hline
1 & * & 0 & 7 & 3 & 3 & 3 & 0 & 3 & 3 & 4 & 0 & 3 \\
2 & & * & 0 & 7 & 6 & 3 & 8 & 3 & 3 & 0 & 8 & 6 \\
3 & & & * & 3 & 3 & 3 & 0 & 3 & 3 & 8 & 0 & 3 \\
5 & & & & * & 6 & 5 & 7 & 5 & 5 & 3 & 7 & 8 \\
7 & & & & & * & 5 & 6 & 5 & 5 & 3 & 6 & 7 \\
8 & & & & & & * & 4 & 8 & 6 & 5 & 4 & 5 \\
9 & & & & & & & * & 4 & 3 & 0 & 8 & 6 \\
11 & & & & & & & & * & 7 & 5 & 4 & 5 \\
13 & & & & & & & & & * & 5 & 3 & 5 \\
14 & & & & & & & & & & * & 0 & 3 \\
17 & & & & & & & & & & & * & 6 \\
19 & & & & & & & & & & & & *
\end{array}$$

studied in [25] (we use also their original names for the vertices). In Figure 3 are reported all the iterations of the SFS algorithm using as initial order of the vertices the reversal of the original labeling of the matrix. At each iteration, the vertices in the blocks are the univisited vertices in the queue.

3.2 Characterization of SFS orderings

In this section we characterize the linear orders returned by the SFS algorithm in terms of a 3-point condition. This characterization applies to any (not necessarily Robinsonian) matrix and it is the analogue of [12, Thm 3.1] for Lex-BFS.

Theorem 3.3. *Given a matrix $A \in \mathcal{S}^n$, an ordering σ of $[n]$ is a SFS ordering of A if and only if the following condition holds:*

$$\begin{array}{l} \textit{For all } x, y, z \in [n] \textit{ such that } A_{xz} > A_{xy} \textit{ and } x <_\sigma y <_\sigma z, \\ \textit{there exists } u \in [n] \textit{ such that } u <_\sigma x \textit{ and } A_{uy} > A_{uz}. \end{array} \tag{4}$$

Proof. ($\Rightarrow$) Suppose σ is a SFS ordering of A. Assume $x <_\sigma y <_\sigma z$ and $A_{xz} > A_{xy}$, but $A_{uz} \geq A_{uy}$ for each $u <_\sigma x$. Assume first that $A_{uz} > A_{uy}$ for some $u <_\sigma x$ and let u be the first such vertex in σ. Then $A_{wz} = A_{wy}$ for each $w <_\sigma u$, and thus y, z are in

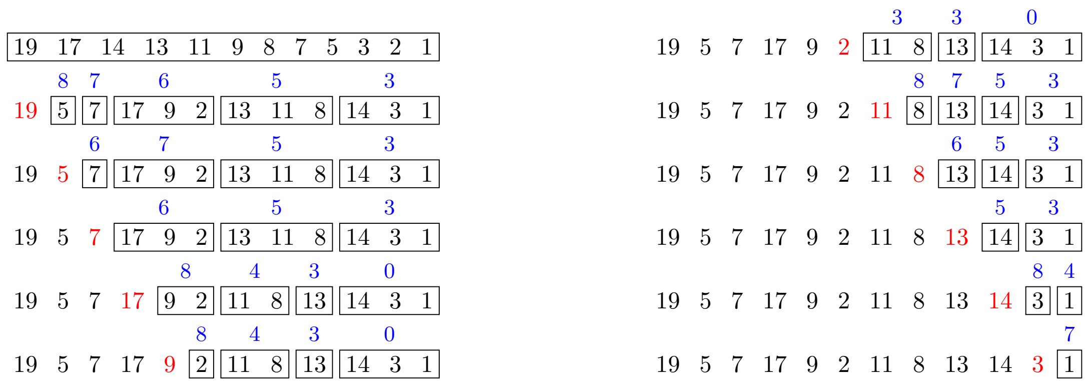

Figure 3: Iterations of SFS algorithm: in red the pivot which is chosen at the current iteration, in blue the similarity between the new pivot and the vertices in the classes of the queue. The first line show the initialization step of the algorithm. The first line on the left shows the initialization step of the algorithm.

the same class of the queue of unvisited vertices when u is chosen as pivot. Therefore, z would be ordered before y in σ when computing the similarity partition of $N(u)$, i.e., we would have $z <_\sigma y$, a contradiction. Hence, one has $A_{uz} = A_{uy}$ for each $u <_\sigma x$. This implies that y, z are in the same class of the queue of unvisited vertices before x is chosen as pivot. Hence, when x is chosen as pivot, as $A_{xz} > A_{xy}$, when computing the similarity partition of $N(x)$ we would get $z <_\sigma y$, which is again a contradiction.

($\Leftarrow$) Assume that the condition (4) of the theorem holds, but σ is not a SFS ordering. Let a denote the first vertex of σ. Let τ be a SFS ordering of A starting at a with the largest possible initial overlap with σ. Say, σ and τ share the same initial order $(a, a_1, \dots, a_r)$ and they differ at the next position. Then we have that $\sigma = (a, a_1, \dots, a_r, y, \dots, z, \dots,)$ and $\tau = (a, a_1, \dots, a_r, z, \dots, y, \dots)$ with $y \neq z$.

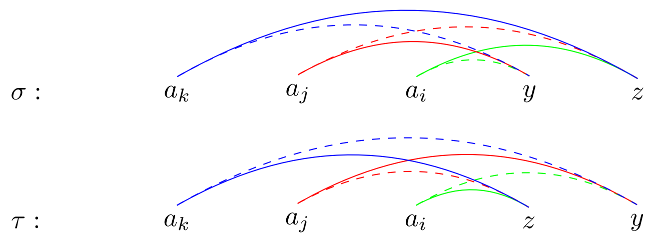

Figure 4: Illustration of the proof of Theorem 3.3 (the dotted lines are strictly smaller than the continuous ones of the same color)

In the SFS ordering τ, the two elements y, z do not lie in the slice of the pivot a_r. Indeed, if y, z would lie in the slice of a_r then one could select y as the next pivot instead of z, which would result in another SFS ordering τ' starting at a and with a larger overlap with σ than τ. Hence, there exists $i \leq r$ such that $A_{a_i z} > A_{a_i y}$. Since $a_i <_\sigma y <_\sigma z$ then applying the condition (4) to σ, we deduce that there exists $j < i$ such that $A_{a_j y} > A_{a_j z}$. Now, we have $a_j <_\tau z <_\tau y$ with $A_{a_j y} > A_{a_j z}$. As τ is a SFS ordering, as we have just shown it must satisfy the condition (4) and thus there must exist an index $k < j$ such that $A_{a_k z} > A_{a_k y}$. Hence, starting from an index $i \leq r$ for which $A_{a_i z} > A_{a_i y}$, we have shown the existence of another index $k < i \leq r$ for which $A_{a_k z} > A_{a_k y}$. Iterating this process, we reach a contradiction. $\square$

One can easily show that if σ is a SFS ordering of V and $S \subseteq V$ is a subset such that any element $x \notin S$ is homogeneous to S, then the restriction $\sigma[S]$ of σ to S is a SFS ordering of $A[S]$. Since, by construction, each vertex before a slice S in σ is homogeneous to S, a direct consequence of Theorem 3.3 is that the restriction of σ to any slice encountered throughout a SFS ordering σ, is a SFS ordering too.

3.3 The Path Avoiding Lemma

In this section we discuss a fundamental lemma which we call the 'Path Avoiding Lemma'. It will play a crucial role throughout the paper and, in particular, for the characterization of anchors. Differently from the analysis in the previous section, where we did not make any assumption on the structure of the matrix A, the Path Avoiding Lemma states some important properties of SFS orderings when the input matrix is Robinsonian.

Before stating this lemma, we need to investigate in more detail the refinement step in the SFS algorithm. An important operation in the Refine task in Algorithm 1 is the splitting procedure of each class of the queue ϕ. The following notion of 'vertex splitting a pair of vertices' is useful to understand it. Consider an order $\sigma = \mathrm{SFS}(A)$ and vertices $x <_\sigma y <_\sigma z$, where $x = p_i$ is the pivot chosen at the ith iteration in Algorithm 1. We say that x *splits* y and z if x is the first pivot for which y and z do not belong to the same class in the queue ordered partition $\phi(p_i)$. Recall that $\phi(p_i)$ denotes the queue of unvisited nodes induced by pivot p_i, i.e., at the end of iteration i (after the refinement step). Hence, saying that y, z are split by x means that y, z belong to a common class B_j of $\phi(p_{i-1})$ and that they belong to distinct classes B_h, B_k of $\phi(p_i)$, where $y \in B_h$, $z \in B_k$ and B_h comes before B_k in $\phi(p_i)$. Equivalently, $x = p_i$ splits y and z if $A_{xy} > A_{xz}$ and $A_{uy} = A_{uz}$ for all $u <_\sigma p_i$.

Then, we say that two vertices $y <_\sigma z$ are *split* in σ if they are split by some vertex $x <_\sigma y$. When y and z are not split in σ, we say that they are *tied*. In this case, ties must be broken between y and z. In the SFS algorithm we assume that ties are broken arbitrarily. In Section 4 we will see the variation SFS_+ of SFS where ties are broken using a linear order τ given as input together with the matrix A. The following lemma will be used as base case for proving the Path Avoiding Lemma.

Lemma 3.4. *Assume that $A \in \mathcal{S}^n$ is a Robinsonian matrix and let $\sigma = \mathrm{SFS}(A)$. Assume that $x <_\sigma y <_\sigma z$ and that there exists a Robinson ordering π of A such that $x <_\pi z <_\pi y$. Then y and z are not split in σ by any vertex $u \leq_\sigma x$. That is, $A_{uy} = A_{uz}$ for all $u \leq_\sigma x$.*

Proof. We first show that y, z are not split by any vertex w occurring before x in σ. Suppose, for contradiction, that y, z are split by a vertex $w <_\sigma x$. Hence, $A_{wy} > A_{wz}$. This implies $z <_\pi w$ for, otherwise, $w <_\pi z <_\pi y$ would imply $A_{wy} \leq A_{wz}$, a contradiction. Hence we have $w <_\sigma x <_\sigma z$ and $x <_\pi z <_\pi w$. Because π is a Robinson ordering, we get $A_{wz} \geq A_{wx}$ and thus $A_{wy} > A_{wz} \geq A_{wx}$. Therefore, the quadruple (w, x, y, z) satisfies the following properties (a)-(d): (a) $w <_\sigma x <_\sigma y <_\sigma z$, (b) $x <_\pi z <_\pi w$ for some Robinson ordering π, (c) w is the pivot splitting y, z, and (d) $A_{wy} > A_{wx}, A_{wz}$ Call any quadruple satisfying (a)-(d) a *bad quadruple.*

We now show that if (w, x, y, z) is a bad quadruple then there exists $u <_\sigma w$ for which (u, w, x, z) is also a bad quadruple. Hence, iterating we will get a contradiction. We now proceed to show the existence of $u <_\sigma w$ for which (u, w, x, z) is also a bad quadruple. Since $A_{wx} < A_{wy}$, the vertices x, y are already split before w becomes a pivot; otherwise, if they would belong to the same class when w is chosen as new pivot, then

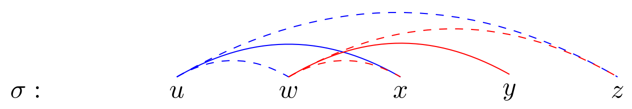

Figure 5: Illustrating the proof of Theorem 3.4: (w, x, y, z) and (u, w, x, z) are bad quadruples (the dotted lines indicate similarities that are strictly smaller than the continuous ones of the same color)

we would get $y <_\sigma x$. Let $u = p_i$ the pivot splitting x, y, i.e., $u <_\sigma w$ and $A_{ux} > A_{uy}$. Thus x, y belong to the same class (say) $B \in \phi(p_{i-1})$ when u is chosen as new pivot at iteration i, but in different classes of $\phi(p_i)$. Since w is the pivot splitting y, z and $u <_\sigma w$, it follows that y, z belong to the same class when u is chosen as pivot, and thus $x, y, z \in B$. Therefore u is also the pivot splitting x and z and thus $A_{ux} > A_{uy} = A_{uz}$. In turn this implies that $u <_\pi z$ for, otherwise, $x <_\pi z <_\pi u$ would imply $A_{ux} \leq A_{uz}$, a contradiction. Therefore, $u <_\pi z <_\pi w$ and by definition of Robinson ordering we have $A_{uw} \leq A_{uz}$ and, as $A_{ux} > A_{uz}$, this implies that $A_{uw} < A_{ux}$. Summarizing, we have shown that the quadruple (u, w, x, z) is bad since it satisfies the conditions (a)-(d): (a) $u <_\sigma w <_\sigma x <_\sigma z$, (b) $w <_{\pi^{-1}} z <_{\pi^{-1}} u$ for the Robinson ordering π^{-1}, (c) u splits x and z, and (d) $A_{ux} > A_{uw}, A_{uz}$. Thus we have shown that there cannot exist a bad quadruple and therefore that y, z are not split by any vertex w appearing before x in σ.

We now conclude the proof of the lemma by showing that y, z are also not split by x. For this, we need to show that $A_{xz} = A_{xy}$. Suppose for contradiction that $A_{xz} \neq A_{xy}$. As $x <_\pi z <_\pi y$, it can only be that $A_{xz} > A_{xy}$. Let $x = p_i$, i.e., x is the pivot chosen at iteration i of Algorithm 1. Since we have just shown that y, z are not split before x, then at the iteration i when x is chosen as pivot, we would order $z <_\sigma y$ as $A_{xz} > A_{xy}$, which is a contradiction because $y <_\sigma z$ by assumption. □

A first direct consequence of Lemma 3.4 is the following.

Corollary 3.5. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix, let $\sigma = SFS(A)$, and consider distinct elements $x, y, z \in V$ such that $x <_\sigma y <_\sigma z$. The following holds:*

(i) $A_{xy} \geq \min\{A_{xz}, A_{yz}\}$.

(ii) If $x <_\pi z <_\pi y$ for some Robinson ordering π, then the path $P = (x, z)$ does not avoid y.

Proof. (i) Assume, for contradiction, that $A_{xy} < \min\{A_{xz}, A_{yz}\}$. Pick a Robinson ordering π of A such that $x <_\pi y$. Then we must have $x <_\pi z <_\pi y$. Indeed, if $x <_\pi y <_\pi z$ then we would have $A_{xy} \geq A_{xz}$, and if $z <_\pi x <_\pi y$ we would have $A_{xy} \geq A_{yz}$, leading in both cases to a contradiction. Applying Lemma 3.4, we conclude that $A_{xy} = A_{xz}$, contradicting our assumption that $A_{xy} < A_{xz}$.

(ii) If (x, z) avoids y then $A_{xz} > \min\{A_{xy}, A_{yz}\}$, where $\min\{A_{xy}, A_{yz}\} = A_{xy}$ since $x <_\pi z <_\pi y$. Hence this contradicts Lemma 3.4. □

Note that the above result is the analogue of the 'P_3-rule' for chordal graphs in [12, Thm 3.12], which claims that, for any distinct $x, y, z \in V$ such that $x <_\sigma y <_\sigma z$ while $x <_\pi z <_\pi y$ for some Robinson ordering π, the path (x, z) does not avoid y. The next lemma strengthens the result of Corollary 3.5 (ii), by showing that there cannot exist *any* path from x to z avoiding y and appearing fully before z in σ. We will refer to Lemma 3.6 below as the 'Path Avoiding Lemma', also abbreviated as (PAL) for ease of reference in the rest of the paper.

Lemma 3.6 (Path Avoiding Lemma (PAL)). *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix and let $\sigma = \mathrm{SFS}(A)$. Consider distinct elements $x, y, z \in V$ such that $x <_\sigma y <_\sigma z$. If $x <_\pi z <_\pi y$ for some Robinson ordering π, then there does not exist a path $P = (x, u_1, \ldots, u_k, z)$ from x to z avoiding y and such that $u_1, \ldots, u_k <_\sigma z$.*

Proof. The proof is by induction on the length $|P| = k+2$ of the path P. The base case is $|P| = 2$, i.e., $P = (x, z)$, which is settled by Corollary 3.5. Assume then, for contradiction, that there exists a path $P = (x, u_1, \ldots, u_k, z)$ from x to z avoiding y with $u_1, \ldots, u_k <_\sigma z$ and $|P| \geq 3$, i.e., $k \geq 1$. Let us call a path Q *short* if it is shorter than P, i.e., if $|Q| < |P|$. By the induction assumption, we know that the following holds:

$$\begin{array}{l}\text{If } u <_\sigma v <_\sigma w \text{ and } u <_\tau w <_\tau v \text{ for some Robinson ordering } \tau,\\ \text{then no short path } Q = (u, v_1, \ldots, v_r, w) \text{ from } u \text{ to } w \text{ avoiding } v\\ \text{and with } v_1, \ldots, v_r <_\sigma w \text{ exists.}\end{array} \tag{5}$$

Set $u_0 = x$ and $u_{k+1} = z$. As P avoids y, the following relations hold:

$$A_{u_{i-1}u_i} > \min\{A_{yu_{i-1}}, A_{yu_i}\} \quad \text{for all } i \in [k+1]. \tag{6}$$

Recall that since $x <_\sigma y <_\sigma z$ and $x <_\pi z <_\pi y$, then in view of Lemma 3.4 we have $A_{xy} = A_{xz}$. Furthermore, we know that $u_1, \ldots, u_k <_\sigma z$ by assumption. In order to conclude the proof, we use the following claim.

Claim 2. *$u_i <_\pi x$ and $y <_\sigma u_i$ for each $i \in [k]$.*

Proof. The proof is by induction on $i \geq 1$. For $i = 1$ we have to show that

$$u_1 <_\pi x \text{ and } y <_\sigma u_1. \tag{7}$$

We first show that $u_1 <_\pi x$. Suppose this is not the case and $x <_\pi u_1$. Recall that in view of (6) for $i = 1$ we have $A_{xu_1} > \min\{A_{yx}, A_{yu_1}\}$ and thus the path (x, u_1) avoids y. Hence, since $x <_\pi y$ and y cannot appear between x and u_1 in any Robinson ordering in view of Lemma 2.4, it must also be that $u_1 <_\pi y$. We then have two possibilities, depending whether u_1 comes before or after z in π.

(i) Assume first that u_1 appears before z in π. Then we have $x <_\pi u_1 <_\pi z <_\pi y$. We discuss where can u_1 appear in σ. If $u_1 <_\sigma y$ then we have $u_1 <_\sigma y <_\sigma z$, $u_1 <_\pi z <_\pi y$, and $(u_1, \ldots, u_k, z)$ is a short path from u_1 to z avoiding y with $u_2, \ldots, u_k <_\sigma z$, which contradicts (5). Hence, $y <_\sigma u_1$ in which case we have $x <_\sigma y <_\sigma u_1$, $x <_\pi u_1 <_\pi y$, and (x, u_1) is a short path from u_1 to z avoiding y, which contradicts again (5).

(ii) Assume now that u_1 appears after z in π. Then we have $x <_\pi z <_\pi u_1 <_\pi y$. By (6) applied to $i = 1$ and using the Robinson ordering π, we then have that $A_{u_1x} > \min\{A_{yx}, A_{yu_1}\} = A_{yx}$. Recall that $A_{xy} = A_{xz}$. Then $A_{u_1x} > A_{xz}$. On the other hand, by the Robinson property of π, $A_{xu_1} \leq A_{xz}$, yielding a contradiction.

Therefore we have shown that $u_1 <_\pi x$. Finally, we show that $y <_\sigma u_1$. Suppose not, i.e., $u_1 <_\sigma y$. Then we would have $u_1 <_\sigma y <_\sigma z$ and, as just shown, $u_1 <_\pi z <_\pi y$, while $(u_1, \ldots, u_k, z)$ is a short path from u_1 to z avoiding y with $u_2, \ldots, u_k <_\sigma z$. This contradicts (5) and thus shows $y <_\sigma u_1$, which concludes the proof for the base case $i = 1$.

Assume now that $i \geq 2$ and that $u_j <_\pi x$ and $y <_\sigma u_j$ for all $1 \leq j \leq i-1$ by induction. We show that also $u_i <_\pi x$ and $y <_\sigma u_i$. First we show $u_i <_\pi x$. Suppose, for the sake of contradiction, that $x <_\pi u_i$. Recall that in view of (6) the path $(u_i, \dots, u_k, z)$ is a path from u_i to z avoiding y with $u_{i+1}, \dots, u_k <_\sigma z$. Hence, since $z <_\pi y$ in view of Lemma 2.4 it must be also $u_1 <_\pi y$, because y cannot appear between z and u_1 in any Robinson ordering. We then have two possibilities to discuss, depending whether u_i comes before or after z in π.

(i) Assume that u_i appears before z in π. Then $u_1, \dots, u_{i-1} <_\pi x <_\pi u_i <_\pi z <_\pi y$. First we claim that $y <_\sigma u_i$. Indeed, if by contradiction $u_i <_\sigma y$, then we would have: $u_i <_\sigma y <_\sigma z$ and $u_i <_\pi z <_\pi y$, while $(u_i, \dots, u_k, z)$ is a short path from u_i to z avoiding y with $u_{i+1}, \dots, u_k <_\sigma z$, contradicting (5).

   Hence, $y <_\sigma u_i$ holds. Recall that $y <_\sigma u_j$ for $j \in [i-1]$ by induction. Hence, for $j = i-1$ we have $y <_\sigma u_{i-1}$. To recap, we are in the case $u_{i-1} <_\pi x <_\pi u_i <_\pi z <_\pi y$ and we have shown that $x <_\sigma y <_\sigma u_i, u_{i-1} <_\sigma z$.

   We thus have $y <_\sigma u_{i-1} <_\sigma z$ and $y <_{\pi^{-1}} z <_{\pi^{-1}} u_{i-1}$. Then, in view of Lemma 3.4, one must have $A_{yu_{i-1}} = A_{yz}$. From the Robinson ordering we obtain that it holds $A_{yz} \geq A_{xy} \geq A_{yu_{i-1}} = A_{yz}$ and therefore we get the equality $A_{yz} = A_{xy}$. Analogously, because $x <_\sigma y <_\sigma u_i$ and $x <_\pi u_i <_\pi y$, by Lemma 3.4 we obtain $A_{xy} = A_{xu_i}$. Hence, we have

$$A_{yu_{i-1}} = A_{yz} = A_{xy} = A_{xu_i} \tag{8}$$

   Finally, using relation (6) we get:

$$A_{u_{i-1}u_i} > \min\{A_{yu_{i-1}}, A_{yu_i}\} = A_{yu_{i-1}}. \tag{9}$$

   In view of (8), the right hand side in (9) is $A_{yu_{i-1}} = A_{xu_i}$. On the other hand, as $u_{i-1} <_\pi x <_\pi u_i$ in the Robinson ordering π, then $A_{yu_{i-1}} = A_{xu_i} \geq A_{u_{i-1}u_i}$, which contradicts (9). Hence u_i cannot appear before z in π.

(ii) Assume u_i appears after z in π. Then $u_1, \dots, u_{i-1} <_\pi x <_\pi z <_\pi u_i <_\pi y$. Observe that the path $(x, u_1, \dots, u_{i-1}, z)$ is a short path from x to z with $u_1, \dots, u_{i-1} <_\sigma z$ and thus it cannot avoid y, otherwise we would contradict (5). Since the path $(x, u_1, \dots u_{i-1})$ avoids y (as it is a subpath of P), it follows that the path (u_{i-1}, z) does not avoid y. Hence $A_{u_{i-1}z} \leq \min\{A_{yu_{i-1}}, A_{yz}\}$ which, using the Robinson ordering π, in turn implies $A_{u_{i-1}z} = A_{yu_{i-1}}$. Then, using relation (6), we get: $A_{u_{i-1}u_i} > \min\{A_{yu_{i-1}}, A_{yu_i}\} = A_{yu_{i-1}}$. Now combining with $A_{yu_{i-1}} = A_{u_{i-1}z}$, we get $A_{u_{i-1}u_i} > A_{u_{i-1}z}$ which is a contradiction, since from the Robinson ordering π one must have $A_{u_{i-1}u_i} \leq A_{u_{i-1}z}$. Therefore we have shown also that u_i cannot appear after z in π.

In summary we have shown that $u_i <_\pi x$ as desired. Finally we show that $y <_\sigma u_i$. Indeed, if $u_i <_\sigma y$ then we would have: $u_i <_\sigma y <_\sigma z$ and $u_i <_\pi z <_\pi y$, while $(u_i, \dots, u_k, z)$ is a short path from u_i to z avoiding y with $u_{i+1}, \dots, u_k <_\sigma z$, which contradicts (5). This concludes the proof of the claim. □

We can now conclude the proof of Lemma 3.6. By Claim 2 we have the following relations for any $i \in [k]$: $x <_\sigma y <_\sigma u_i <_\sigma z$ and $y <_{\pi^{-1}} z <_{\pi^{-1}} x <_{\pi^{-1}} u_i$. By

Lemma 3.4, this implies $A_{yu_i} = A_{yz}$ for all $i \in [k]$ which, using the Robinson ordering π, in turn implies $A_{yu_i} = A_{yz} = A_{yx}$. Now, use relation (6) for $i = k+1$ to get the inequality $A_{u_k z} > \min\{A_{yu_k}, A_{yz}\} = A_{yu_k}$. Recall that in view of Lemma 3.4, we have that $A_{xy} = A_{xz}$. Then as $A_{yu_i} = A_{yx}$ for all i, the right hand side is equal to $A_{yu_k} = A_{xz}$ while, using the Robinson ordering π, the left hand side satisfies $A_{u_k z} \leq A_{xz}$, which yields a contradiction. This concludes the proof of the lemma. □

3.4 End points of SFS orderings

In this section we show some fundamental properties of SFS orderings, using the results in Section 3.3. First we show that if A is Robinsonian then the last vertex of a SFS ordering of A is an anchor of A. We will see later in Corollary 4.3 that conversely any anchor can be obtained as end point of a SFS ordering.

Theorem 3.7. *Let A be a Robinsonian matrix and let $\sigma = \mathrm{SFS}(A)$. Then the last vertex of σ is an anchor of A.*

Proof. Let z be the last vertex of σ; we show that z is an anchor of A. In view of Theorem 2.7 it suffices to show that z is valid. Suppose for contradiction that, for some $x \neq y \in V \setminus \{z\}$, there exist a path P from x to z avoiding y and a path Q from y to z avoiding x. We may assume without loss of generality that $x <_\sigma y <_\sigma z$. Moreover, let π be a Robinson ordering of A such that $x <_\pi z$. Then, in view of Lemma 2.4, we must have $x <_\pi z <_\pi y$, since y must come either before or after both x and z (because of the path P) and x must come before or after both y and z (because of the path Q). As z is the last vertex, then $P <_\sigma z$ and thus we get a contradiction with Lemma 3.6 (PAL). □

The above result is the analogue of [12, Thm 4.5] for Lex-BFS applied to interval graphs (see [11] for more results about end points of Lex-BFS orderings). We now introduce the concept of 'good SFS'.

Definition 3.8 (**Good SFS ordering**). *We say that a SFS ordering σ of A is* good *if σ starts with a vertex which is the end vertex of some SFS ordering.*

Note that the analogous definition in [12] for Lex-BFS is stronger, as it requires the first vertex of each slice to be an end vertex of the slice itself. However, in our discussion we do not need such a strong definition and the above notion of good SFS will suffice to show the overall correctness of the multisweep algorithm. In the case when A is Robinsonian, in view of Theorem 3.7 (and Corollary 4.3 below), σ is a good SFS ordering precisely when it starts with an anchor of A. For good SFS orderings we have the following stronger result for their end points.

Theorem 3.9. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix and let σ be a good SFS ordering whose first vertex is a and whose last vertex is b. Then a, b are opposite anchors of A.*

Proof. By assumption, σ is a good SFS ordering and thus its first vertex a is an anchor of A. In view of Theorem 3.7, its last vertex b is also an anchor of A. Suppose, for the sake of contradiction, that a and b are not opposite anchors of A. Then, in view of Theorem 2.8, there exists a vertex $x \in V$ and a path P from a to b such that P avoids x. Let π be a Robinson ordering of A starting with a (which exists since a is an anchor of A). Using Lemma 2.4 applied to the path P, we can conclude that x cannot appear between a and b in any Robinson ordering, and thus we must have $a <_\pi b <_\pi x$. But then, using Lemma 3.6 (PAL), there cannot exist a path from a to b avoiding x and appearing before b in σ, which contradicts the existence of P. □

4 The SFS$_+$ algorithm

In this section we introduce the SFS$_+$ algorithm. This is a variant of the standard SFS algorithm, and it is the analogue of the variant Lex-BFS+ of Lex-BFS introduced by Simon [30] in the study of multisweep algorithms for interval graphs. The algorithm SFS$_+$ will be the main ingredient in our multisweep algorithm for the recognition of Robinsonian matrices. It takes as input a matrix A and a linear order σ, and it returns another linear order σ_+. After describing SFS$_+$, we will first present its main properties, most importantly the fact that the SFS$_+$ algorithm 'flips anchors' when applied to a Robinsonian matrix A and a good SFS order σ: if σ starts at a and ends at b, then σ_+ starts at b and ends at a. We will also introduce the useful concept of 'similarity layers' of a matrix, which will play a crucial role in the correctness analysis of our multisweep SFS-based algorithm.

4.1 Description of the SFS$_+$ algorithm

Consider again the SFS algorithm as described in Algorithm 1 in Section 3. The first main task is selecting the new pivot. In case of ties, as done at Line 4 of Algorithm 1, the ties are broken arbitrarily (choosing any vertex in the slice S). We now introduce a variant of SFS(A), which we denote by SFS$_+(A, \sigma)$. It takes as input a matrix $A \in \mathcal{S}^n$ and a linear order σ of V, and it returns a new linear order σ_+ of V. In the SFS$_+$ algorithm, the input linear order σ is used to break ties at Line 4 in Algorithm 1. Specifically, among the vertices in the slice S of the current iteration, we choose the vertex appearing last in σ. Notice that a SFS$_+$ ordering is still a SFS ordering and thus it satisfies all the properties discussed in Section 3.

Algorithm 2: SFS$_+(A, \sigma)$

```
input: a matrix $A \in \mathcal{S}^n$ and a linear order $\sigma$ of $[n]$
output: a linear order $\sigma_+$ of $[n]$

1  $\phi = (V)$
2  for $i = 1, \dots, n$ do
3      $S$ is the first class of $\phi$
4      choose $p$ as the vertex in $S$ appearing last in $\sigma$
5      $\sigma_+(p) = i$
6      remove $p$ from $\phi$
7      $N(p)$ is the set of vertices $y \in \phi$ with $A_{py} > 0$
8      $\psi_p$ is the similarity partition of $N(p)$ with respect to $p$
9      $\phi =$ Refine $(\phi, \psi_p)$
10 return: $\sigma_+$
```

If A is a Robinsonian matrix and the input linear order σ is a SFS ordering, then the SFS$_+$ ordering σ_+ has some important properties. In fact, since in the beginning of the SFS algorithm all the vertices are contained in the 'universal' slice (i.e., the full ground set V), the order σ_+ starts with the last vertex of σ, which in view of Lemma 3.7 is an anchor of A. Therefore, in this case, σ_+ is a good SFS ordering by construction. Furthermore, in view of Theorem 3.9, when A is Robinsonian then the first and last vertices of σ_+ are opposite anchors of A.

If the input linear order σ is a good SFS ordering, then we have an even stronger property: the end points of σ_+ are the end points of σ but in reversed order. We call this the 'anchors flipping property', which is shown in the next theorem. This property will be crucial in Section 5 when studying the properties of the multisweep algorithm.

Theorem 4.1 (**Anchors flipping property**). *Let $A \in \mathcal{S}^n$ be a Robinsonian similarity, let σ be a good SFS ordering of A and $\sigma_+ = \mathrm{SFS}_+(A, \sigma)$. Suppose that σ starts with a and ends with b. Then σ_+ starts with b and ends with a.*

Proof. By definition of the SFS$_+$ algorithm, the returned order σ_+ starts with the last vertex b of σ. Hence, we only have to show that a appears last in σ^+. Suppose, for the sake of contradiction, that a is not last in σ^+ and let instead y be the vertex appearing last in σ^+. Then we have $a <_\sigma y <_\sigma b$ and $b <_{\sigma_+} a <_{\sigma_+} y$. This implies that y and a must be split in σ^+. Indeed, if y and a would be tied in σ^+ then, as we use σ to break ties and as $a <_\sigma y$, the vertex y would be placed before a in σ^+, a contradiction. Let thus $x <_{\sigma_+} a$ be the pivot splitting a and y in σ^+, so that $A_{xa} > A_{xy}$. Then we have:

$$A_{xa} > \min\{A_{xy}, A_{ya}\}. \tag{10}$$

Hence the path $P = (x, a)$ avoids y. As b is the first vertex of σ^+, we have:

$$b <_{\sigma_+} x <_{\sigma_+} a <_{\sigma_+} y.$$

In view of Theorem 3.9 applied to σ, we know that a and b are opposite anchors of A. Therefore, there exists a Robinson ordering π starting with a and ending with b. In view of (10) and using Lemma 2.4, y cannot appear between a and x in any Robinson ordering and therefore we can conclude:

$$a <_\pi x <_\pi y <_\pi b. \tag{11}$$

Consider now σ. We have that $a <_\sigma y <_\sigma b$. Where can x appear in σ? Suppose $y <_\sigma x$. Then we would have $a <_\sigma y <_\sigma x$ and $a <_\pi x <_\pi y$, and in view of Lemma 3.6 (PAL) there cannot exist a path from a to x avoiding y and appearing before x in σ, which is a contradiction as the path $P = (x, a)$ avoids y in view of (10). Hence, we must have:

$$a <_\sigma x <_\sigma y <_\sigma b. \tag{12}$$

Therefore, starting from the pair (a, y) satisfying $a <_\sigma y$ and $a <_{\sigma_+} y$, we have constructed a new pair (x, y) satisfying $x <_\sigma y$ and $x <_{\sigma_+} y$, with $x <_{\sigma_+} a$. Iterating this construction we are going to get an infinite sequence of such pairs, yielding a contradiction. $\square$

The flipping property of anchors is the analogue of [12, Thm 4.6] for Lex-BFS. An important consequence of this property is that, if the linear order σ given as input is a Robinson ordering of A, then $\sigma_+ = \mathrm{SFS}_+(a, \sigma)$ is equal to σ^{-1}, the reversed order of σ.

Lemma 4.2. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix and let σ, τ be two SFS orderings of A. The following holds:*

(i) If $x <_\tau y <_\tau z$ and $z <_\sigma y <_\sigma x$ then the triple (x, y, z) is Robinson.

(ii) If τ is a Robinson ordering of A and $\sigma = \mathrm{SFS}_+(A, \tau)$, then $\sigma = \tau^{-1}$.

Proof. (i) Suppose for contradiction that the triple (x, y, z) is not Robinson. Then we have $A_{xz} > \min\{A_{xy}, A_{yz}\}$, and thus the path (x, z) avoids y. Let π be a Robinson ordering of A with (say) $x <_\pi y$. In view of Lemma 2.4, y cannot appear between x and z in any Robinson ordering and therefore we have $x <_\pi z <_\pi y$ or $z <_\pi x <_\pi y$. In both cases we get a contradiction with Lemma 3.6 (PAL) since $x <_\tau y <_\tau z$ and $z <_\sigma y <_\sigma x$.

(ii) Say τ starts at b and ends at a. Then σ starts at a. Assume that $\sigma \neq \pi^{-1}$. Let $(a = x_0, x_1, \ldots, x_k)$ be the longest initial segment of σ whose reverse $(x_k, \ldots, x_1, a)$ is the final segment of τ, with $k \geq 0$. Let y be the successor of x_k in σ. Then y is not the predecessor of x_k in τ (by maximality of k). Let z be the predecessor of x_k in τ. Then $a <_\sigma x_1 <_\sigma \ldots <_\sigma x_k <_\sigma y <_\sigma z$ and $y <_\tau z <_\tau x_k <_\tau \ldots <_\tau x_1 <_\tau a$. Hence, y, z cannot be tied in σ (otherwise we would choose z before y in σ as $y <_\tau z$). Therefore, there must exist a vertex $x <_\sigma y$ such that $A_{xy} > A_{xz}$. Hence, $x = x_i$ for some $0 \leq i \leq k$ and thus $y <_\tau z <_\tau x$. As τ is a Robinson ordering, then $A_{xy} \leq A_{xz}$, a contradiction. □

In other words, in a multisweep algorithm, every triple of vertices appearing in reversed order in two distinct sweeps is Robinson and, once a given sweep is a Robinson ordering, the next sweep will remain a Robinson ordering (precisely the reversed order). As an important application, one can check if a given order σ is Robinson simply by computing the order $\sigma_+ = \mathrm{SFS}_+(A, \sigma)$, and checking if it is the reversed of σ, hence without checking the Robinson property for all the entries of the matrix. This is analogous to a similar feature shown in [14] for their multisweep algorithm to recognize cocomparability graphs.

As a direct application of Lemma 4.2 combined with Theorem 3.7, we obtain the following characterization for anchors.

Corollary 4.3. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix. A vertex is an anchor of A if and only if it is the end point of a SFS ordering of A.*

4.2 Similarity layers

In this section we introduce the notion of 'similarity layer structure' for a matrix $A \in \mathcal{S}^n$ and an element $a \in V$ (then called the *root*), which we will use later to analyze properties of the multisweep algorithm.

Specifically, we define the following collection $\mathcal{L} = (L_0, L_1, \ldots, L_r)$ of subsets of V, whose members are called the *(similarity) layers of A rooted at a*, where $L_0 = \{a\}$ and the next layers L_i are the subsets of V defined recursively as follows:

$$L_i = \{y \notin L_0 \cup \cdots \cup L_{i-1} : A_{xy} \geq A_{xz} \ \forall x \in L_0 \cup \cdots \cup L_{i-1}, \ \forall z \notin L_0 \cup \cdots \cup L_{i-1}\}. \quad (13)$$

Note that this notion of similarity layers can be seen as a refinement of the notion of BFS layers for graphs, which are obtained by layering the nodes according to their distance to the root. Hence, the two concepts are similar but different. We first show that this layer structure defines a partition of V when A is a Robinsonian matrix and the root a is an anchor of A.

Lemma 4.4. *Assume that $A \in \mathcal{S}^n$ is a Robinsonian matrix and that $a \in V$ is an anchor of A. Consider the similarity layer structure $\mathcal{L} = (L_0 = \{a\}, L_1, \dots, L_r)$ of A rooted at a, as defined in (13), where r is the smallest index such that $L_{r+1} = \emptyset$. The following holds:*

(i) If $y \in L_i, z \notin L_0 \cup \ldots \cup L_i$ with $i \geq 1$, then there exists a path P from a to y avoiding z. Moreover, any path of the form $P = (a, a_1, \dots, a_i = y)$, where $a_l \in L_l$ for $1 \leq l \leq i$, avoids z.

(ii) $V = L_0 \cup L_1 \cup \ldots \cup L_r$.

Proof. (i) Using the definition of the layers in (13) we obtain that $A_{aa_1} > A_{az}$ and $A_{a_1a_2} > A_{a_1z}, \dots, A_{a_{i-1}y} > A_{a_{i-1}z}$, which shows that the path $(a, a_1, \dots, a_{i-1}, a_i = y)$ avoids z.

(ii) Suppose $L_0, L_1, \dots, L_r \neq \emptyset$, $L_{r+1} = \emptyset$, but $V \neq U := L_0 \cup \ldots \cup L_r$. Consider an element $z_1 \in V \setminus U$. As $z_1 \notin L_r$ (since this set is empty) there exist elements $x_1 \in U$ and $z_2 \notin U$ such that $A_{x_1z_1} < A_{x_1z_2}$. Analogously, as $z_2 \notin L_r$ there exist elements $x_2 \in U, z_3 \notin U$ such that $A_{x_2z_2} < A_{x_2z_3}$. Iterating we find elements $x_i \in U$, $z_i \notin U$ for all $i \geq 1$ such that $A_{x_iz_i} < A_{x_iz_{i+1}}$ for all i. At some step one must find one of the previously selected elements z_i, i.e., $z_j = z_i$ for some $i < j$.

As a is an anchor of A, there exists a Robinson ordering π of A starting at a. We first claim that $x_i <_\pi z_j$ for all i, j. This is clear if $x_i = a$. Otherwise, as $x_i \in U$ and $z_j \notin U$, it follows from (i) that there is a path from a to x_i avoiding z_j, which in view of Lemma 2.4 implies that $a \leq_\pi x_i <_\pi z_j$. Next we claim that $z_{i+1} <_\pi z_i$. Since $x_i <_\pi z_i$ and $A_{x_iz_i} < A_{x_iz_{i+1}}$, then (x_i, z_{i+i}) avoids z_i and in view of Lemma 2.4 it must be indeed $z_{i+1} <_\pi z_i$. Summarizing we have shown that $z_{i+1} <_\pi z_i <_\pi \ldots <_\pi z_1$ for all i, which contradicts the fact that two of the z_i's should coincide. □

Intuitively, each layer L_i will correspond to some slices of a SFS algorithm starting at a. As we see below, there is some compatibility between the layer structure $\mathcal{L}$ rooted at a with any Robinson ordering π and any good SFS ordering σ starting at a.

Lemma 4.5. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix and a is an anchor of A. Let σ be a good SFS ordering of A starting at a and let π be a Robinson ordering of A starting at a. Then the similarity layer structure $\mathcal{L} = (L_0 = \{a\}, \dots, L_r)$ of A rooted at a is compatible with both π and σ. That is,*

$$L_0 <_\pi L_1 <_\pi \ldots <_\pi L_r,$$
$$L_0 <_\sigma L_1 <_\sigma \ldots <_\sigma L_r.$$

Proof. Let $x \in L_i$ and $y \in L_j$ with $i < j$; we show that $x <_\pi y$ and $x <_\sigma y$. This is clear if $i = 0$, i.e., if $x = a$. Suppose now $i \geq 1$. Then, by Lemma 4.4, there exists a path from a to x avoiding y. This implies that $a <_\pi x <_\pi y$, as y cannot appear between a and x in any Robinson ordering in view of Lemma 2.4 and since π starts with a. Furthermore, if $a <_\sigma y <_\sigma x$ then we would get a contradiction with Lemma 3.6 (PAL). Hence $a <_\sigma x <_\sigma y$ holds, as desired. □

Furthermore, the following inequalities hold among the entries of A indexed by elements in different layers.

Lemma 4.6. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix and a is an anchor of A. Let $\mathcal{L} = (L_0 = \{a\}, L_1, \dots, L_r)$ be the similarity layer structure of A rooted at a. For each $u \in L_i, x, y \in L_j$ and $z \notin L_0 \cup L_1 \cup \dots \cup L_j$ with $0 \le i < j$ the following inequalities hold:*

$$A_{xy} \ge A_{ux} = A_{uy} \ge A_{uz}.$$

Furthermore, if $x \in L_j, z \notin L_0 \cup L_1 \cup \cdots \cup L_j$, then there exists $u \in L_0 \cup L_1 \cup \cdots \cup L_{j-1}$ such that $A_{ux} > A_{uz}$.

Proof. The inequalities $A_{ux} = A_{uy} > A_{uz}$ follow from the definition of the layers in (13). Suppose now that $A_{xy} < A_{ux} = A_{uy}$. Then u must appear between x and y in any Robinson ordering π, since $x <_\pi y <_\pi u$ implies $A_{xy} \ge A_{ux}$ and $y <_\pi x <_\pi u$ implies $A_{xy} \ge A_{uy}$. But in view of Lemma 4.5, if π is a Robinson ordering starting at a then $u <_\pi x$ and $u <_\pi y$, so we get a contradiction. □

As an application of Lemma 4.6, it is easy to verify that if A is the adjacency matrix of a connected graph G, then each layer is a clique of G.

We now show a 'flipping property' of the similarity layers with respect to a good SFS ordering σ starting at the root and the next sweep $\sigma_+ = \text{SFS}_+(A, \sigma)$. Namely we show that the orders of the layers are reversed beween σ and σ_+, i.e., $L_i <_\sigma L_j$ and $L_j <_{\sigma_+} L_i$ for all $i < j$.

Theorem 4.7 (Layers flipping property). *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix and $a \in V$ be an anchor of A. Let $\mathcal{L} = (L_0 = \{a\}, \dots, L_r)$ be the similarity layer structure of A rooted at a, let σ be a good SFS ordering of A starting at a and let $\sigma^+ = \text{SFS}_+(A, \sigma)$. If $x \in L_i$, $y \in L_j$ with $0 \le i < j \le r$ then $y <_{\sigma_+} x$.*

Proof. Let $x \in L_i$, $y \in L_j$ with $i < j$. Assume for contradiction that $x <_{\sigma_+} y$. By Lemma 4.5, we know that $\mathcal{L}$ is compatible with σ and thus $x <_\sigma y$. As $x <_{\sigma_+} y$ and $x <_\sigma y$, we deduce that x, y are not tied in σ^+. Hence there exists $x_1 <_{\sigma_+} x$ such that $A_{x_1x} > A_{x_1y}$. Let L_ℓ denote the layer of $\mathcal{L}$ containing x_1. We claim that $\ell < j$. Indeed, if $\ell = j$ then x_1, y are in the same layer and, by Lemma 4.6, it must be $A_{x_1y} \ge A_{x_1x} = A_{xy}$ which is impossible, because $A_{x_1x} > A_{x_1y}$. Assume now that $\ell > j$. By Lemma 4.5, if π is a Robinson ordering starting at a, then we would get $x <_\pi y <_\pi x_1$, which implies $A_{x_1y} \ge A_{x_1x}$, again a contradiction. Therefore, we have $x_1 \in L_\ell$ with $\ell < j$. Recall that $x_1 <_{\sigma_+} y$. Hence, starting with the pair (x, y) which satisfies $x \in L_i$, $y \in L_j$ with $i < j$ and $x <_{\sigma_+} y$, we have constructed another pair (x_1, y) satisfying $x_1 \in L_l$, $y \in L_j$ with $l < j$ and $x_1 <_{\sigma_+} y$. As $x_1 <_+ x$, iterating this construction we will reach a contradiction. □

5 The multisweep algorithm

We now introduce our new SFS-based multisweep algorithm and we show that in at most $n-1$ sweeps it permits to recognize whether a given matrix of size n is Robinsonian. This is the main result of our paper, which we will prove in this section. First in Section 5.1 we will describe the algorithm and its main features. Then in Section 5.2 we introduce the notion of '3-good sweep' which plays a crucial role in the correctness proof and we investigate its properties. Finally, in Section 5.3 we complete the proof of correctness of the multisweep algorithm.

5.1 Description of the multisweep algorithm

Our multisweep algorithm consists of computing successive SFS orderings of a given nonnegative matrix $A \in \mathcal{S}^n$. The first sweep is SFS(A), whose aim is to find an anchor of A. Each subsequent sweep is computed with the SFS$_+$ algorithm using the linear order returned by the preceding sweep to break ties (as in Algorithm 2). As it starts with the end point of the preceding sweep which is an anchor of A, each subsequent sweep is therefore a good SFS ordering of A (in the case when A is Robinsonian). The algorithm terminates either if a Robinson ordering has been found (in which case it certifies that A is Robinsonian), or if the $(n-1)$th sweep is not Robinson (in which case it certifies that A is not Robinsonian). The complete algorithm is reported below.

```
Algorithm 3: Robinson(A)
   input: a matrix $A \in \mathcal{S}^n$
   output: a Robinson ordering $\pi$ of $A$, or stating that $A$ is not Robinsonian

1  $\sigma_0 = \text{SFS}(A)$
2  for $i = 1, \dots n-2$ do
3      $\sigma_i = \text{SFS}_+(A, \sigma_{i-1})$
4      if $\sigma_i$ is Robinson then
5          return: $\pi = \sigma_i$
6  return: '$A$ is NOT Robinsonian'
```

We first report below some examples for $n = 4, 5, 6$ where the number of sweeps needed in Algorithm 3 is tight, i.e., equal to $n-1$. Note that for $n = 3$ it is clear that two sweeps may be needed and always suffice.

Consider the following (Robinson) matrices:

$$A_4 = \begin{array}{c|cccc} & a & b & c & d \\ \hline a & * & 1 & 1 & 0 \\ b & & * & 2 & 1 \\ c & & & * & 2 \\ d & & & & * \end{array}, \quad A_5 = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & * & 2 & 2 & 0 & 0 \\ b & & * & 2 & 1 & 1 \\ c & & & * & 2 & 1 \\ d & & & & * & 1 \\ e & & & & & * \end{array}, \quad A_6 = \begin{array}{c|cccccc} & a & b & c & d & e & f \\ \hline a & * & 1 & 1 & 1 & 1 & 0 \\ b & & * & 2 & 2 & 1 & 1 \\ c & & & * & 2 & 2 & 2 \\ d & & & & * & 3 & 2 \\ e & & & & & * & 2 \\ f & & & & & & * \end{array}.$$

Consider first the matrix A_4. Then $\sigma_0 = (b, c, d, a)$ is a valid SFS ordering of A_4 and $\sigma_1 = (a, c, b, d)$, which is not a Robinson ordering. However, $\sigma_2 = (d, c, b, a)$ is a Robinson ordering of A_4.

Consider now the matrix A_5. Then $\sigma_0 = (c, d, b, a, e)$ is a valid SFS ordering of A_5, with $\sigma_1 = (e, b, c, d, a)$ and $\sigma_2 = (a, c, b, d, e)$, which is not a Robinson ordering. However, $\sigma_3 = (e, d, c, b, a)$ is a Robinson ordering of A_5.

Consider finally the matrix A_6. Then $\sigma_0 = (b, d, c, e, f, a)$ is a valid SFS ordering of A_6, with $\sigma_1 = (a, e, d, c, b, f)$, $\sigma_2 = (f, c, d, e, b, a)$ and $\sigma_3 = (a, b, d, c, e, f)$ which is not a Robinson ordering. However, $\sigma_4 = (f, e, d, c, b, a)$ is a Robinson ordering of A_6.

As already mentioned earlier, the SFS algorithm applied to binary matrices reduces to Lex-BFS. As a warm-up we now show that our SFS multisweep algorithm terminates in three sweeps to recognize whether a binary matrix A is Robinsonian. As a binary matrix

A is Robinsonian if and only if the corresponding graph is a unit interval graph [26], this is coherent with the fact that one can recognize unit interval graphs in three sweeps of Lex-BFS [8, Thm 9]. Hence we have a new proof for this result, which has similarities but yet differs from the original proof in [8].

Theorem 5.1. *Let G be a connected graph and let A be its adjacency matrix. Consider the orders $\sigma_0 = \mathrm{SFS}(A)$, $\sigma_1 = \mathrm{SFS}_+(A, \sigma_0)$ and $\sigma_2 = \mathrm{SFS}_+(A, \sigma_1)$. Then G is a unit interval graph (i.e., A is Robinsonian) if and only if σ_2 is a Robinson ordering of A.*

Proof. Clearly, if σ_2 is Robinson then A is Robinsonian. Assume now that A is Robinsonian; we show that σ_2 is Robinson. Suppose, for contradiction, that there exists a triple $x <_{\sigma_2} y <_{\sigma_2} z$ which is not Robinson, i.e., $A_{xz} > \min\{A_{xy}, A_{yz}\}$. Then the path (x, z) avoids y and thus, in view of Lemma 3.6 (PAL), in any Robinson ordering π one cannot have $x <_\pi z <_\pi y$. We may assume without loss of generality that $z <_\pi x <_\pi y$ in some Robinson ordering π. Because A is a binary matrix, then $A_{xz} = 1$, $A_{yz} = 0$ and thus $\{x, z\} \in E, \{y, z\} \notin E$. By construction, σ_1 is a good SFS ordering of A starting (say) at the anchor a. Let $\mathcal{L} = \{L_0, L_1 \dots, L_r\}$ be the similarity layer structure of A rooted at a. By Lemma 4.5, we know that $\mathcal{L}$ is compatible with σ_1, i.e., $a <_{\sigma_1} L_1 <_{\sigma_1} \dots <_{\sigma_1} L_r$. Using Theorem 4.7 we obtain that $L_r <_{\sigma_2} L_{r-1} <_{\sigma_2} \dots <_{\sigma_2} L_1 <_{\sigma_2} a$. Moreover, using Lemma 4.6 and the fact that G is connected, it is easy to see that each layer L_i is a clique of G. Hence, y, z cannot be in the same layer of $\mathcal{L}$, as $\{y, z\} \notin E$. Since $y <_{\sigma_2} z$, it follows that $z \in L_i, y \in L_j$ with $i < j$ and thus $z <_{\sigma_1} y$. Say $x \in L_h$. One cannot have $h < j$ since this would contradict $x <_{\sigma_2} y$. If $h = j$ then $x, y \in L_j$ and thus $A_{zx} = A_{zy}$ by definition of the layers, contradicting the fact that $A_{xz} = 1$, $A_{yz} = 0$. Hence one must have $j < h$. Then $z \in L_i$, $y \in L_j$, $x \in L_h$ with $i < j < h$ and thus $z <_{\sigma_1} y <_{\sigma_1} x$. Now we get a contradiction with Lemma 3.6 (PAL), as $z <_\pi x <_\pi y$ and the path (x, z) avoids y. □

We now formulate our main result, namely that the SFS multisweep algorithm terminates in at most $n - 1$ steps to recognize whether an $n \times n$ matrix is Robinsonian.

Theorem 5.2. *Let $A \in \mathcal{S}^n$ and let $\sigma_0 = \mathrm{SFS}(A)$, $\sigma_i = \mathrm{SFS}_+(A, \sigma_{i-1})$ for $i \geq 1$ be the successive sweeps returned by Algorithm 3. Then A is a Robinsonian matrix if and only if σ_{n-2} is a Robinson ordering of A.*

We will give the full proof of Theorem 5.2 in Section 5.3 below. What we need to show is that if A is Robinsonian then the order σ_{n-2} in Algorithm 3 is a Robinson ordering of A. We now give a rough sketch of the strategy which we will use to prove this result. The proof will use induction on the size n of the matrix A.

As was shown earlier, the sweep σ_1 is a good SFS ordering of A with end points (say) a and b, and all subsequent sweeps have the same end points (flipping their order at each sweep) in view of Theorem 4.1. A first key ingredient will be to show that if we delete both end points a and b and set $S = V \setminus \{a, b\}$, then the induced order $\sigma_3[S]$ is a good SFS ordering of the principal submatrix $A[S]$. A second crucial ingredient will be to show that the induced order $\sigma_{n-2}[S]$ can be obtained with the multisweep algorithm applied to $A[S]$ starting from $\sigma_3[S]$. This will enable us to apply the induction assumption and to conclude that $\sigma_{n-2}[S]$ is a Robinson ordering of $A[S]$. Hence all triples (x, y, z) in σ_{n-2} that are contained in $S = V \setminus \{a, b\}$ are Robinson. The last step is to show that all triples (x, y, z) in σ_{n-2} that contain a or b are also Robinson.

As we see in the above sketch, the sweep σ_3 plays a special role. It is obtained by applying three sweeps of SFS_+ starting from the good SFS ordering σ_1. For this reason we call it a *3-good SFS ordering.* We introduce and investigate in detail this notion of '3-good SFS ordering' in Section 5.2 below.

5.2 3-good SFS orderings

Consider a Robinsonian matrix $A \in \mathcal{S}^n$. Recall that a SFS ordering τ of A is said to be *good* if its first vertex is an anchor of A (see Section 4.1). We now introduce the notion of 3-good SFS ordering. A linear order τ is called a *3-good SFS ordering* of A if there exists a good SFS ordering τ' of A such that, if we set $\tau'' = \mathrm{SFS}_+(A, \tau')$, then $\tau = \mathrm{SFS}_+(A, \tau'')$ holds. In other words, a 3-good SFS ordering is obtained by performing three consecutive good sweeps. Of course any 3-good SFS ordering is also a good SFS ordering. Furthermore, if we consider Algorithm 3, then any sweep σ_i with $i \geq 3$ is a 3-good SFS ordering by construction. First we report the following flipping property of layers which follows as a direct application of Theorem 4.7.

Corollary 5.3. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix. Let τ' be a good SFS ordering of A, $\tau'' = \mathrm{SFS}_+(A, \tau')$ and $\tau = \mathrm{SFS}_+(A, \tau'')$. Let $\mathcal{L} = \{L_0, \dots, L_r\}$ be the similarity layer structure of A rooted at the first vertex of τ. If $x \in L_i$, $y \in L_j$ with $i < j$ then $y <_{\tau''} x$.*

We now show some important properties of 3-good SFS orderings, that we will use in the proof of correctness of the multisweep algorithm. First we show that some triples in a 3-good SFS ordering can be shown to be Robinson.

Lemma 5.4. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix. Let τ be a 3-good SFS ordering starting at a and ending at b. Let $\mathcal{L} = \{L_0 = \{a\}, L_1, \dots, L_r\}$ be the similarity layer structure of A rooted at a. Then the following holds:*

(i) If $x <_\tau y <_\tau z$ and (x, y, z) is not Robinson, then $x, y, z \in L_i$ with $1 \leq i \leq r$.

(ii) Every triple (a, x, y) with $x <_\tau y$ is Robinson.

(iii) Every triple (x, y, b) with $x <_\tau y$ is Robinson.

Proof. Let τ' be a good SFS order such that $\tau'' = \mathrm{SFS}_+(A, \tau')$, $\tau = \mathrm{SFS}_+(A, \tau'')$. Let $\mathcal{L}'' = \{L''_0 = \{b\}, L''_1, \dots\}$ denote the similarity layer structure of A rooted at b, which is compatible with τ''.

(i) Let $x <_\tau y <_\tau z$ such that x, y, z do not all belong to the same layer of $\mathcal{L}$ and assume that (x, y, z) is not Robinson. Then $A_{xz} > \min\{A_{xy}, A_{yz}\}$ and the path (x, z) avoids y. Let π be a Robinson ordering and assume, without loss of generality, that $x <_\pi z$. Then, since (x, z) avoids y, in view of Lemma 2.4 y cannot appear between x and z in any Robinson ordering. If y appears after z in π then we have $x <_\pi z <_\pi y$ and $x <_\tau y <_\tau z$, and we get a contradiction with Lemma 3.6 (PAL) as there cannot exists a path from x to z avoiding y. Therefore $y <_\pi x <_\pi z$ and thus $A_{xz} > \min\{A_{xy}, A_{yz}\} = A_{yz}$. In view of Lemma 4.5, x, y, z do not belong to three distinct layers of $\mathcal{L}$ (since otherwise (x, y, z) would be Robinson). Moreover, one cannot have $x \in L_i$ and $y, z \in L_j$ with $i < j$ (since this would imply $A_{xy} = A_{xz} \leq A_{yz}$, a contradiction). Hence we must have $x, y \in L_i$ and $z \in L_j$ with $i < j$.

Consider now τ''; applying Corollary 5.3, we derive that $z <_{\tau''} x, y$. Moreover, we cannot have that $z <_{\tau''} y <_{\tau''} x$, since we would get a contradiction with Lemma 3.6

(PAL) as $z <_{\pi^{-1}} x <_{\pi^{-1}} y$ and the path (x,z) avoids y. Hence we have $z <_{\tau''} x <_{\tau''} y$. Summarizing, the triple (x,y,z) satisfies the properties:

$$x, y \in L_i, \quad z \in L_j, \quad x <_\tau y <_\tau z, \quad x <_{\tau''} y, \quad y <_\pi x <_\pi z. \tag{14}$$

We will now show that the properties in (14) (together with the inequality $A_{xz} > A_{yz}$) permit to find an element $x_1 <_\tau x$ for which the triple (x_1, y, z) again satisfies the properties of (14), replacing x by x_1. Then, iterating this construction leads to a contradiction.

We now proceed to show the existence of such an element x_1. As $x <_{\tau''} y$ and $x <_\tau y$, x, y are not tied in τ and thus there exists $x_1 <_\tau x$ such that

$$A_{x_1x} > A_{x_1y}.$$

This implies $x_1 \in L_i$ (recall Lemma 4.6). Moreover, the path (x_1, x, z) avoids y, since $A_{x_1x} > A_{x_1y}$ and $A_{xz} > A_{yz}$. By construction we have: $x_1 <_\tau x <_\tau y <_\tau z$. We claim that

$$y <_\pi x_1 <_\pi z.$$

Indeed, if $x_1 <_\pi y$, then $x_1 <_\pi y <_\pi x$ and thus $A_{x_1x} \leq A_{x_1y}$, a contradiction. Moreover, if $z <_\pi x_1$ then $y <_\pi x <_\pi z <_\pi x_1$, which implies $A_{x_1z} \geq A_{x_1x} > A_{x_1y}$ and thus the triple (x_1, y, z) is not Robinson. Then $A_{x_1z} > \min\{A_{x_1y}, A_{yz}\}$ and the path (x_1, z) avoids y. Now, as $x_1 <_\tau y <_\tau z$ and $x_1 <_{\pi^{-1}} z <_{\pi^{-1}} y$, we get a contradiction with Lemma 3.6 (PAL). So we have shown that $y <_\pi x_1 <_\pi z$.

Next we claim that $x_1 <_{\tau''} y$. Indeed, if $y <_{\tau''} x_1$ then $z <_{\tau''} y <_{\tau''} x_1$, which together with $z <_{\pi^{-1}} x_1 <_{\pi^{-1}} y$ and the fact that the path (x_1, x, z) avoids y, contradicts Lemma 3.6 (PAL). Hence we have shown that the triple (x_1, y, z) satisfies (14), which concludes the proof of *(i)*.

(ii) follows directly from *(i)*, since any triple containing a is not contained in a unique layer, and thus it must be Robinson.

(iii) Assume for contradiction that (x, y, b) is not Robinson for some $x <_\tau y$, i.e., $A_{bx} > \min\{A_{by}, A_{xy}\}$. Then the path (b, x) avoids y. If π is a Robinson ordering ending at b (which exists since b is an anchor) then we must have $y <_\pi x <_\pi b$ because, in view of Lemma 2.4, y cannot appear between x and b in any Robinson ordering. Hence, $A_{bx} > A_{by}$. Since τ'' is compatible with $\mathcal{L}''$ which is rooted at b, we must have $b <_{\tau''} x <_{\tau''} y$ and moreover x, y belong to distinct layers of $\mathcal{L}''$. Thus $x \in L''_i, y \in L''_j$ with $i < j$ which, in view of Theorem 4.7, implies $y <_\tau x$, a contradiction. $\square$

As a first direct application of Lemma 5.4(i), we can conclude that the multisweep algorithm terminates in at most four steps when applied to a matrix A whose similarity layers rooted at the end vertex of the first sweep σ_0 all have cardinality at most 2.

Consider a 3-good SFS ordering τ of a Robinsonian matrix A with end points a and b and consider the induced order $\tau[S]$ of the submatrix $A[S]$ indexed by the subset $S = V \setminus \{a, b\}$. In the next lemmas we show some properties of $\tau[S]$. First, we show that $\tau[S]$ is a good SFS ordering of $A[S]$ (Lemma 5.6). Second, we show that applying the SFS$_+$ algorithm to τ and then deleting a and b yields the same order as applying the SFS$_+$ algorithm to the induced order $\tau[S]$ (Lemma 5.7). These properties will be used in the induction step for the proof of correctness of the multisweep algorithm in the next section. We start with showing a 'flipping property' of the second smallest element of τ.

Lemma 5.5. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix. Let τ' be a good SFS ordering of A, $\tau'' = \mathrm{SFS}_+(A, \tau')$ and $\tau = \mathrm{SFS}_+(A, \tau'')$. Let a be the first vertex of τ. Then the successor a_1 of a in τ is the predecessor of a in τ''.*

Proof. As before, $\mathcal{L} = \{L_0 = \{a\}, L_1, \ldots, L_r\}$ is the layer structure of A rooted at a, which is compatible with τ. The slice of a in τ is precisely the first layer L_1 in $\mathcal{L}$. By definition, a_1 is the element of L_1 coming last in τ''. By the flipping property in Corollary 5.3, we know that the layer L_1 comes last but one in τ'', just before the layer $L_0 = \{a\}$. Then a_1 is the element of L_1 appearing last in τ'', and thus it coincides with the predecessor of a in τ''. $\square$

Lemma 5.6. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix. Let τ be a 3-good SFS ordering of A with end points a and b and set $S = V \setminus \{a, b\}$. Then $\tau[S]$ is a good SFS ordering of $A[S]$.*

Proof. Say that a is the first element of τ and that b is its last element. Consider the similarity layer structure rooted at a, i.e., $\mathcal{L} = (L_0, L_1, \ldots, L_r)$, which is compatible with τ. First we show that $\tau[S]$ is a SFS ordering of $A[S]$. For this consider elements $x, y, z \in S$ such that $A_{xz} > A_{xy}$. Then (x, y, z) is not Robinson and thus $x, y, z \in L_i$ with $i \geq 1$ in view of Lemma 5.4. As τ is a SFS ordering, then in view of Theorem 3.3 there exists $u <_\tau x$ such that $A_{uy} > A_{uz}$. We have $u \neq a$ (since $u = a$ would imply $A_{uy} = A_{uz}$) and thus $u \in S$. This shows that $\tau[S]$ is a SFS ordering of $A[S]$. Finally $\tau[S]$ is good since, in view of Lemma 5.5, it starts at a_1, the successor of a in τ, which is an anchor of $A[V \setminus \{a\}]$ (and thus also of $A[S]$) in view of Theorem 3.7. $\square$

Lemma 5.7. *Assume $A \in \mathcal{S}^n$ is a Robinsonian matrix. Let τ be a 3-good SFS ordering order with end points a and b and $\tau^+ = \mathrm{SFS}_+(A, \tau)$. Then $\tau^+[S] = \mathrm{SFS}_+(A[S], \tau[S])$ with $S = V \setminus \{a, b\}$.*

Proof. Say b is the first element of τ and a be its last element. Then a is the first element of τ^+ and b is its last element (Theorem 4.1). Let consider the similarity layer structure $\mathcal{L} = (L_0 = \{a\}, L_1, \ldots, L_r)$ of A rooted at a, which is compatible with τ^+ (and thus we denote here by $\mathcal{L}^+$).

Set $\sigma = \mathrm{SFS}_+(A[S], \tau[S])$. Let a_1 the predecessor of a in τ. As τ^+ is clearly also a 3-good SFS ordering then, in view of Lemma 5.5, a_1 is the successor of a in τ^+ and thus both $\tau^+[S]$ and σ start at a_1. Assume that σ and $\tau^+[S]$ agree on their first p elements $a_1, \ldots, a_p$, but not at the next $(p+1)$th element. That is, $\tau^+[S] = (a_1, \ldots, a_p, x, \ldots, y, \ldots)$, while $\sigma = (a_1, \ldots, a_p, y, \ldots, x, \ldots)$, where x, y are distinct elements. We distinguish three cases.

Assume first that x, y are tied in τ^+ (and thus in σ too). Then one must have $y <_\tau x$ (to place x before y in $\tau^+[S]$) and $x <_\tau y$ (to place y before x in σ), a contradiction.

Assume now that x, y are not tied in τ^+, but they are tied in σ. Then $A_{ax} > A_{ay}$. Hence, since the similarity layer structure $\mathcal{L}$ of A is rooted at a, then we have $x \in L_j$, $y \in L_k$ for some $j < k$. This implies $y <_\tau x$ (by Corollary 5.3) and thus, since x, y are tied in σ, one would place x before y in σ, a contradiction.

Assume finally that x, y are not tied in τ^+ and also not in σ. Let a_j be the pivot splitting x and y in σ so that $A_{a_j y} > A_{a_j x}$, with $1 \leq j \leq p$. We claim that a is the pivot splitting x and y in $\tau^+[S]$. For this, suppose that a_i is the pivot splitting x and y in $\tau^+[S]$ for some $1 \leq i \leq p$, so that $A_{a_i x} > A_{a_i y}$ and $i \neq j$. It is now easy to see that $i > j$ would imply $y <_{\tau^+} x$, while $i < j$ would imply $x <_\sigma y$, a contradiction in both cases. Hence, a is the pivot splitting x, y in $\tau_+[S]$ and thus $A_{ax} > A_{ay}$. Then, as $\mathcal{L}^+$ is the similarity layer structure of A rooted at a, x and y belong to distinct layers of $\mathcal{L}^+$. Moreover, $a_j <_{\tau_+} x <_{\tau_+} y$ and the triple (a_j, x, y) is not Robinson. As τ^+ is a 3-good SFS ordering, we can apply Lemma 5.4 and conclude that a_j, x, y must belong to a common layer of $\mathcal{L}$, which contradicts the fact that x, y belong to distinct layers of $\mathcal{L}^+$. $\square$

5.3 Proof of correctness of the multisweep algorithm

We can finally put all ingredients together and show the correctness of our multisweep algorithm. We show the following result, which implies directly Theorem 5.2.

Theorem 5.8. *Let $A \in \mathcal{S}^n$ be a Robinsonian matrix, let τ_1 be a good SFS ordering of A and let $\tau_i = \mathrm{SFS}_+(A, \tau_{i-1})$ for $i \geq 2$. Then τ_{n-2} is a Robinson ordering of A.*

Proof. The proof is by induction on the size n of A. For $n < 3$ there is nothing to prove and for $n = 3$ the result holds trivially. Hence, suppose $n \geq 4$. Then, by the induction assumption, we know that the following holds:

> If σ_1 is a good SFS ordering of a Robinsonian matrix $A' \in \mathcal{S}^k$ with $k \leq n-1$ and $\sigma_i = \mathrm{SFS}_+(A', \sigma_{i-1})$ for $i \geq 2$, then σ_{k-2} is a Robinson ordering of A'.

Suppose τ_1 starts with a and ends with b. By Lemma 4.1, the end points of any τ_i with $i \geq 2$ are a and b (flipped at every consecutive sweep). For any $i \geq 3$, τ_i is a 3-good SFS ordering of A. Hence, setting $S = V \setminus \{a, b\}$, in view of Lemma 5.7, we obtain that $\tau_{i+1}[S] = \mathrm{SFS}_+(A[S], \tau_i[S])$ for each $i \geq 3$.

Consider the order $\sigma_1 := \tau_3[S]$ and the successive sweeps $\sigma_i = \mathrm{SFS}_+(A[S], \sigma_{i-1})$ ($i \geq 2$) returned by the multisweep algorithm applied to $A[S]$ starting from σ_1. As τ_3 is a 3-good SFS ordering of A, in view of Lemma 5.6 we know that σ_1 is a good SFS ordering of $A[S]$. Hence, using the induction assumption applied to $A[S]$ and σ_1, we can conclude that the sweep $\sigma_{|S|-2} = \sigma_{n-4}$ (returned by the multisweep algorithm applied to $A[S]$ with σ_1 as first sweep) is a Robinson ordering of $A[S]$.

We now observe that equality $\tau_{i+2}[S] = \sigma_i$ holds for all $i \geq 1$, using induction on $i \geq 1$. This is true for $i = 1$ by the definition of σ_1. Inductively, if $\tau_{i+2}[S] = \sigma_i$ then $\tau_{i+3}[S] = \mathrm{SFS}_+(A[S], \tau_{i+2}[S]) = \mathrm{SFS}_+(A[S], \sigma_i) = \sigma_{i+1}$. Hence, we can conclude that $\tau_{n-2}[S] = \sigma_{n-4}$ is a Robinson ordering of $A[S]$. Finally, using with Lemma 5.4, we can conclude that all triples (x, y, z) in τ_{n-2} that contain a or b are Robinson. Therefore we have shown that τ_{n-2} is a Robinson ordering of A, which concludes the proof. $\square$

In other words, starting with a good SFS ordering of a Robinsonian matrix $A \in \mathcal{S}^n$, after at most $n-2$ sweeps we find a Robinson ordering of A. Finally, we can now prove Theorem 5.2, since the last vertex of the first sweep σ_0 in Algorithm 3 is an anchor of A (Theorem 3.7) and thus the second sweep σ_1 is a good SFS ordering.

Hence, if $A \in \mathcal{S}^n$ is a Robinsonian matrix, in view of Theorem 5.8, the multisweep algorithm returns a Robinson ordering in at most $n-2$ sweeps starting from σ_1, and thus in at most $n-1$ sweeps counting also the initialization sweep σ_0.

6 Complexity

In this section we discuss the complexity of the SFS algorithm. Throughout we assume that $A \in \mathcal{S}^n$ is a nonnegative symmetric matrix, given as adjacency list of an undirected weighted graph $G = (V = [n], E)$. So G is the support graph of A, whose edges are the pairs $\{x, y\}$ such that $A_{xy} > 0$ with edge weight A_{xy}, and $N(x) = \{y \in V : A_{xy} > 0\}$ is the neighborhood of $x \in V$. We assume that each vertex $x \in V = [n]$ is linked to the list of vertex/weight pairs (y, A_{xy}) for its neighbors $y \in N(x)$ and we let m denote the number of nonzero entries of A.

Theorem 6.1. *The SFS algorithm (Algorithm 1) applied to an $n \times n$ symmetric nonnegative matrix with m nonzero entries runs in $O(n + m \log n)$ time.*

Proof. As in [8] For Lex-BFS, we may assume that we are given an initial order τ of V and that the vertices and their neighborhoods are ordered according to τ (in increasing order). This assumption is useful also for the discussion of the implementation of SFS$_+$.

In order to run Algorithm 1, we need to update the queue ϕ consisting of the unvisited vertices at each iteration. The update consists in computing the similarity partition ψ_p with respect to the current pivot p and then refining ϕ by ψ_p.

To maintain the priority among the unvisited vertices, the queue $\phi = (B_1, \ldots, B_p)$ is stored in a linked list, whose elements are the classes $B_1, \ldots, B_p$. Moreover each vertex has a pointer to the class B_i containing it and a pointer to its position in the class, which are updated throughout the algorithm. This data structure permits constant time insertion and deletion of a vertex in ϕ.

Initially, the queue ϕ has only one class, namely the full set V. At an iteration of Algorithm 1, there are three main tasks to be performed: choose the next pivot, compute the similarity partition ψ_p and refine the queue ϕ by ψ_p.

(1) Choose the new pivot $p = p_i$. Since in Algorithm 1 the choice of the new pivot is arbitrary in case of ties, we will choose the first vertex of the first block in ϕ. This operation can be done in constant time. We then remove p from the queue ϕ of unvisited vertices and we update the queue ϕ by deleting p from the class B_1.

(2) Compute the similarity partition $\psi_p = (C_1, \ldots, C_s)$ of the set $N_\phi(p)$ with respect to $p = p_i$ (as defined in Definition 3.2). Here $N_\phi(p) = N(p) \cap \phi$ denotes the set of unvisited vertices in the neighborhood $N(p)$ of p. First we order the vertices y in $N_\phi(p)$ for nonincreasing values of their similarities A_{py} with respect to p, which can be done in in $O(|N_\phi(p)| \log |N_\phi(p)|)$ time using a sorting algorithm. Then we create the similarity partition $\psi_p = (C_1, \ldots, C_s)$ simply by passing through the elements in $N_\phi(p)$ in the order of nonincreasing similarities to p which has just been found. This task can be done in $O(|N_\phi(p)|)$ time. Finally we order the elements in each class C_j (increasingly) according to τ, which can be done in $O(|N_\phi(p)| \log |N_\phi(p)|)$. So we have constructed the ordered partition $\psi_p = (C_1, \ldots, C_s)$ of $N_\phi(p)$ as a linked list, where all classes of ψ_p are ordered according to τ. To conclude, the overall complexity of this second task is bounded by $O(|N_\phi(p)| \log |N_\phi(p)|)$.

(3) The last task is to refine $\phi = (B_1, \ldots, B_p)$ by $\psi_p = (C_1, \ldots, C_s)$ (as defined in Definition 3.1). In order to obtain the new queue of unvisited vertices we proceed as follows: starting from $j = 1$, for each class C_j of ψ, we simply remove each vertex of C_j from its corresponding class (say) B_i in ϕ and we place it in a new class B_i' which we position immediately before B_i in ϕ. Since both C_j and B_i are ordered according to τ, the initial order τ in the new block B_i' is preserved. Using the above described data structure, such tasks can be performed in $O(|C_j|)$. Once a vertex is relocated in ϕ, its pointers to the corresponding block and position in ϕ are updated accordingly. Hence this last task can be performed in time $O(\sum_{j=1}^{s} |C_j|) = O(|N_\phi(p)|)$.

Recall that at iteration i we set $p = p_i$. Then the complexity at the ith iteration is $O(1 + |N_\phi(p_i)| \log |N_\phi(p_i)|)$. Since we repeat the above three tasks for each vertex, the overall complexity is $O(\sum_{i=1}^{n} (1 + N_\phi(p_i)| \log |N_\phi(p_i)|)) = O(n + m \log n)$. $\square$

Using the same data structure as above, we can show that the SFS_+ algorithm can be implemented in the same running time as the SFS algorithm.

Theorem 6.2. *The* SFS_+ *algorithm (Algorithm 2) applied to an $n \times n$ symmetric nonnegative matrix with m nonzero entries runs in $O(n + m \log n)$ time.*

Proof. The only difference between the SFS algorithm and the SFS_+ algorithm lies in the tie-breaking rule. In the SFS_+ algorithm, in case of ties we choose as next pivot the vertex in the slice appearing last in the given order σ. We now show that, using the same data structure and assumption as in the proof of Theorem 6.1, this choice can be done in constant time, and thus does not affect the complexity of the SFS_+ algorithm.

Recall that we assumed V to be initially ordered according to a linear order τ. We now select $\tau = \sigma^{-1}$. Then, since we showed that the initial order τ is always preserved in the classes of ϕ throughout the algorithm, we ensure that the first vertex in each slice S is exactly the vertex of S appearing first in τ, i.e., last in σ.

Hence, the only thing we need to discuss is the complexity of reordering A according to τ. This can be done in $O(m + n)$ time as follows. We build a new adjacency list A' where the vertices are ordered according to τ: starting from the vertex appearing first in τ, for each vertex x in τ and for each $y \in N(x)$, we push $\tau(x)$ back in the list of A' corresponding to the neighbors of y. At the end of the process, each neighborhood in A' is then sorted according to τ.

Therefore, the overall complexity of the SFS_+ algorithm is also $O(n + m \log n)$. $\square$

It follows directly from Theorem 6.2 that any SFS multisweep algorithm with k sweeps can be implemented in $O(k(n + m \log n))$. Indeed the only additional tasks we need to do are the following: when we start a new SFS_+ sweep we need to reorder the vertices and their neighborhoods according to the reversal of the previous sweep, and we need to check if the current sweep σ_i is a Robinson ordering, which can be both done in $O(m+n)$ time. Therefore, as the multisweep algorithm (Algorithm 3) needs $k \leq n - 1$ sweeps, it runs in time $O(n^2 + nm \log n)$.

As already mentioned in Section 3, if the matrix has only $0-1$ entries, then there is no need to order the neighborhood $N(p)$ of a given pivot p, because the similarity partition ψ_p has only one class, equal to $N(p)$. For this reason, in this case the SFS algorithm can be implemented in linear time $O(m + n)$. Furthermore, as shown in Theorem 5.1, three sweeps suffice in the multisweep algorithm to find a Robinson ordering. Therefore, if A is a binary matrix, the multisweep algorithm in Algorithm 3 has an overall running time of $O(m + n)$. This is coherent with the fact that in the $0 - 1$ case SFS reduces to Lex-BFS.

When the graph G associated to the matrix A is connected the complexity of SFS and SFS_+ is $O(m \log n)$. Of course we may assume without loss of generality that we are in the connected case since we may deal with the connected components independently. Indeed a matrix A is Robinsonian if and only if the submatrices $A[C]$ are Robinsonian for all connected components C of G, and Robinson orderings of the connected components $A[C]$ can be concatenated to give a Robinson ordering of the full matrix A.

Finally we observe that we may also exploit the potential sparsity induced by the *largest* entries of A. While G is the graph whose edges are the pairs $\{x, y\}$ with entry $A_{xy} > 0$ (where 0 is the smallest possible entry as A is assumed to be nonnegative), we can also consider the graph G' whose edges are the pairs $\{x, y\}$ with entry $A_{xy} < A_{\max}$, where $A_{\max}$ is the largest possible entry of A. Let $N'(p)$ denote the neighborhood of a vertex p in G' and let m' denote the number of entries with $A_{xy} < A_{\max}$. We claim that the SFS (SFS$_+$) algorithm can also be implemented in time $O(n + m' \log n)$.

For this we modify the definition of the similarity partition of a vertex p, which is now a partition of $N'(p)$ (so that the vertices $y \notin N'(p)$ have entry $A_{py} = A_{\max}$) and the refinement of the queue ϕ by it: while we previously build the queue ϕ of unvisited vertices using a 'push-first' strategy (put the vertices with highest similarity first) we now build the queue with a 'push-last' strategy (put the vertices with lowest similarity last).

7 Conclusions

In this paper we have introduced the new search algorithm *Similarity-First Search* (SFS) and its variant SFS$_+$, which are generalizations to weighted graphs of the classical Lex-BFS algorithm and its variant Lex-BFS$_+$. The algorithm is entirely based on the main task of partition refinement, it is conceptually simple and easy to implement. We have shown that a multisweep algorithm can be designed using SFS and SFS$_+$, which permits to recognize if a symmetric $n \times n$ matrix is Robinsonian and if so to return a Robinson ordering after at most $n - 1$ sweeps. We believe that this recognition algorithm is substantially simpler than the other existing algorithms. Moreover, to the best of our knowledge, this is the first work extending multisweep graph search algorithms to the setting of weighted graphs (i.e., matrices).

Our algorithm can also be used to recognize Robinsonian dissimilarities. Recall that a matrix $D \in \mathcal{S}^n$ is a *Robinson dissimilarity matrix* if $D_{xz} \geq \max\{D_{xy}, D_{yz}\}$ for all $1 \leq x < y < z \leq n$, and a *Robinsonian dissimilarity* if its rows and columns can be simultaneously reordered to get a Robinson dissimilarity matrix. Clearly D is a Robinsonian dissimilarity matrix if and only if the matrix $A = -D$ is a Robinsonian similarity matrix. Therefore, one can check whether D is a Robinsonian dissimilarity by applying the SFS-based multisweep algorithm to the matrix A.

Alternatively one may also modify the SFS algorithm so that it can deal directly with dissimilarity matrices. Say D is a nonnegative dissimilarity matrix and G is the corresponding weighted graph with edges the pairs $\{x, y\}$ with $D_{xy} > 0$. Then we can modify the SFS algorithm as follows. First, we now order the vertices in the neighborhood $N(p)$ of a vertex p for *nondecreasing values* of the dissimilarities D_{py} (instead of nonincreasing values of the similarities A_{py} as was the case in SFS). Then we construct the (dis)similarity partition ψ_p of $N(p)$ by grouping the vertices with the same dissimilarity to p, in increasing values of the dissimilarities. Finally, when refining the queue ϕ by ψ_p, we apply a 'push-first' strategy and place the vertices with lowest dissimilarity first. The resulting algorithm, which we name DiSFS, standing for *Dissimilarity-Search First*, has the same running time in $O(n + m \log n)$. Moreover, as explained above at the end of Section 6, it can also be implemented in time $O(n + m' \log n)$, where m' denotes the number of entries of D satisfying $D_{xy} < D_{\max}$ and $D_{\max}$ denotes the largest entry of D. Using DiSFS we can define the analogous multisweep algorithm for recognizing Robinsonian dissimilarities in time $O(n^2 + nm \log n)$ (or $O(n^2 + nm' \log n)$).

It is an open question whether the number of sweeps needed to recognize Robinsonian (dis)similarities can be bounded by a constant number. If this would be the case, then the multisweep algorithm would have running time $O(n + m \log n)$ and thus become (theoretically) competitive with the optimal one in [25]. However, as we have seen in Section 5, there are examples for $n = 4, 5, 6$ where $n - 1$ sweeps are needed. Hence, in order to show a constant number of sweeps, one possibly needs to define another variant of SFS, where ties are broken using the SFS orderings returned by two previous sweeps (and not only one as in the SFS$_+$ variant). This approach has been succesfully applied to Lex-BFS for the recognition of interval graphs in five Lex-BFS sweeps [12], where the last sweep used is the variant Lex-BFS$_*$, which breaks ties using the linear order returned by two previous sweeps. Dusart and Habib [14] conjecture that a similar approach applies to recognize cocomparability graphs with a constant number of sweeps. Investigating whether such an approach applies to Robinsonian matrices will be the subject of future work.

Finally, it will be interesting to investigate whether the new SFS algorithm can be used to study other classes of structured matrices and in the general area of similarity search and clustering analysis.

Acknowledgements

This work was supported by the Marie Curie Initial Training Network "Mixed Integer Nonlinear Optimization" (MINO) grant no. 316647. The authors wish to thank Michel Habib for his useful comments on the complexity of the algorithm. The second author also thanks Mohammed El Kebir for useful discussions.

References

[1] J.E. Atkins, E.G. Boman, and B. Hendrickson. A spectral algorithm for seriation and the consecutive ones problem. *SIAM Journal on Computing*, 28:297–310, 1998.

[2] S.T. Barnard, A. Pothen, and H.D. Simon. A spectral algorithm for envelope reduction of sparse matrices. In *Proceedings of the 1993 ACM/IEEE Conference on Supercomputing*, Supercomputing '93, pages 493–502, 1993.

[3] K.S. Booth and G.S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976.

[4] A. Bretscher, D.G. Corneil, M. Habib, and C. Paul. A simple linear time LexBFS cograph recognition algorithm. *SIAM Journal on Discrete Mathematics*, 22:1277–1296, 2008.

[5] V. Chepoi and B. Fichet. Recognition of Robinsonian dissimilarities. *Journal of Classification*, 14(2):311–325, 1997.

[6] V. Chepoi, B. Fichet, and M. Seston. Seriation in the presence of errors: Np-hardness of l_∞-fitting Robinson structures to dissimilarity matrices. *Journal of Classification*, 26(3):279–296, 2009.

[7] V. Chepoi and M. Seston. Seriation in the presence of errors: A factor 16 approximation algorithm for l_∞-fitting Robinson structures to distances. *Algorithmica*, 59(4):521–568, 2011.

[8] D.G. Corneil. A simple 3-sweep LBFS algorithm for the recognition of unit interval graphs. *Discrete Applied Mathematics*, 138(3):371–379, 2004.

[9] D.G. Corneil. Lexicographic breadth first search - A survey. In J. Hromkovi, M. Nagl, and B. Westfechtel, editors, *Graph-Theoretic Concepts in Computer Science*, volume 3353 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin Heidelberg, 2005.

[10] D.G. Corneil, H. Kim, S. Natarajan, S. Olariu, and A.P. Sprague. Simple linear time recognition of unit interval graphs. *Information Processing Letters*, 55(2):99–104, 1995.

[11] D.G. Corneil, E. Khler, and J.M. Lanlignel. On end-vertices of lexicographic breadth first searches. *Discrete Applied Mathematics*, 158(5):434 – 443, 2010.

[12] D.G. Corneil, S. Olariu, and L. Stewart. The LBFS structure and recognition of interval graphs. *SIAM Journal on Discrete Mathematics*, 23(4):1905–1953, 2009.

[13] P. Crescenzi, D.G. Corneil, J. Dusart, and M. Habib. New trends for graph search. PRIMA Conference in Shanghai, June 2013. available at `http://math.sjtu.edu.cn/conference/Bannai/2013/data/20130629B/slides1.pdf`.

[14] J. Dusart and M. Habib. A new LBFS-based algorithm for cocomparability graph recognition. *Discrete Applied Mathematics*, to appear, 2015.

[15] F. Fogel, A. d' Aspremont, and M. Vojnovic. Serialrank: Spectral ranking using seriation. In *Advances in Neural Information Processing Systems 27*, pages 900–908. 2014.

[16] F. Fogel, R. Jenatton, F. Bach, and A. d'Aspremont. Convex relaxations for permutation problems. *SIAM Journal on Matrix Analysis and Applications*, 36(4):1465–1488, 2015.

[17] J.Y. Goulermas, A. Kostopoulos, and T.Mu. A new measure for analyzing and fusing sequences of objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99), 2015.

[18] M. Habib, R. McConnell, C. Paul, and L. Viennot. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theoretical Computer Science*, 234(12):59–84, 2000.

[19] M. Hahsler, K. Hornik, and C. Buchta. Getting things in order: An introduction to the R package seriation. *Journal of Statistical Software*, 25(1), 2008.

[20] M. Laurent and M. Seminaroti. A Lex-BFS-based recognition algorithm for Robinsonian matrices. In *Algorithms and Complexity: Proceedings of the 9th International Conference CIAC 2015*, volume 9079 of *Lecture Notes in Computer Science*, pages 325–338. Springer-Verlag, 2015.

[21] I. Liiv. Seriation and matrix reordering methods: An historical overview. *Statistical Analysis and Data Mining*, 3(2):70–91, 2010.

[22] P.J. Looges and S. Olariu. Optimal greedy algorithms for indifference graphs. *Computers & Mathematics with Applications*, 25(7):15–25, 1993.

[23] B.G. Mirkin and S.N. Rodin. *Graphs and Genes.* Biomathematics (Springer-Verlag). Springer, 1984.

[24] R. Paige and R.E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.

[25] P. Préa and D. Fortin. An optimal algorithm to recognize Robinsonian dissimilarities. *Journal of Classification*, 31:1–35, 2014.

[26] F.S. Roberts. Indifference graphs. In *Proof Techniques in Graph Theory: Proceedings of the Second Ann Arbor Graph Theory Conference, F. Harary, ed.*, pages 139–146, New York, NY, 1969. Academic Press.

[27] W.S. Robinson. A method for chronologically ordering archaeological deposits. *American Antiquity*, 16(4):293–301, 1951.

[28] D.J. Rose and R.E. Tarjan. Algorithmic aspects of vertex elimination. In *Proceedings of 7th Annual ACM Symposium on Theory of Computing*, STOC '75, pages 245–254. ACM, 1975.

[29] M. Seston. *Dissimilarités de Robinson: Algorithmes de Reconnaissance et d'Approximation.* PhD thesis, Université de la Méditerranée, 2008.

[30] K. Simon. A new simple linear algorithm to recognize interval graphs. In H. Bieri and H. Noltemeier, editors, *Computational geometry-methods, algorithms and applications*, volume 553 of *Lecture Notes in Computer Science*, pages 289–308. Springer Berlin Heidelberg, 1991.