Combinatorial and algorithmic properties of Robinson matrices

Monique Laurent

Joint work with Matteo Seminaroti

Centrum Wiskunde & Informatica (CWI), Amsterdam & Tilburg University

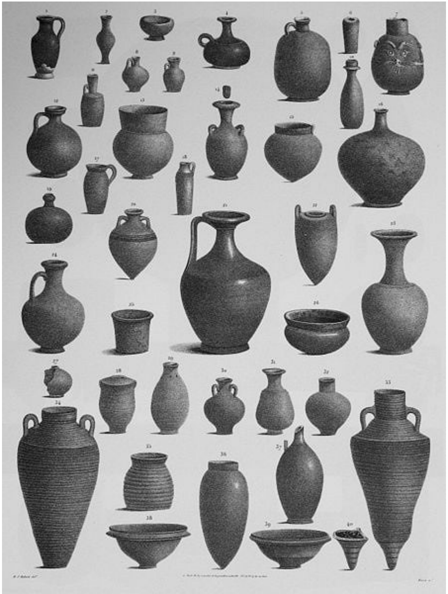
LAGOS 2017 - CIRM, Marseille

Rough plan

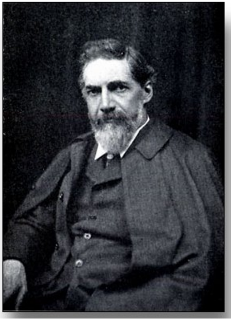
- The seriation problem: Robinson matrices and the spectral algorithm
- Combinatorial algorithms: links to (unit) interval (hyper)graphs
- Classical graph search: Lexicographic Breadth-First Search (Lex-BFS) & unit interval graphs
- New weighted graph search: Similarity-First Search (SFS) & Robinson matrices

The seriation problem

Motivation: Archeology



Sequence dating



Sir William Matthew Flinders Petrie (1853-1942)

Consecutive Ones Property (C1P)

Order the graves chronologically based on the stylistic and technical characteristics of objects (potteries...) found in the sites.

$$\begin{array}{c} \\ G1 \\ G2 \\ G3 \\ G4 \\ G5 \end{array} \begin{array}{c} \begin{array}{cccc} P1 & P2 & P3 & P4 \end{array} \\ \begin{pmatrix} & 1 & & \\ 1 & & 1 & 1 \\ & & 1 & 1 \\ & & 1 & \\ 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Matrix with C1P

P

$$\begin{array}{c} \\ G1 \\ G5 \\ G2 \\ G3 \\ G4 \end{array} \begin{array}{c} \begin{array}{cccc} P1 & P2 & P3 & P4 \end{array} \\ \begin{pmatrix} & 1 & & \\ 1 & 1 & 1 & 1 \\ 1 & & 1 & 1 \\ & & 1 & 1 \\ & & 1 & \end{pmatrix} \end{array}$$

Petrie matrix

ΠP

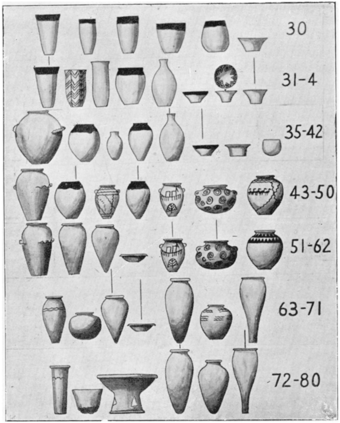


Fig. 1.—Types of pottery of seven successive stages, the sequence dates of each being at the right. In each stage are shown forms which are peculiar to that stage, together with two forms which pass through into an adjacent stage. It will be readily seen how impossible it would be to invert the order of any of these stages without breaking up the links between them. At the left ends of the five lower rows is the wavy-handled type, in its various stages; the degradation of this type was the best clue to the order of the whole period.

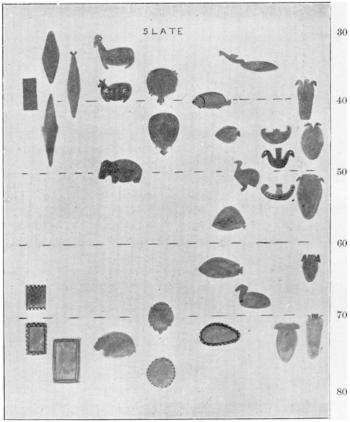


Fig. 2.—The forms of the slate palettes, used for grinding face paint, are very varied. The rhomb is the earliest type, but died out by 37, except in rude forms, which lasted till 47. Quadrupeds are well worked at first, but become rough by 40, and rarely recognisable later on. Fishes and turtles begin at 36, become rude by about 50, and were ovals and discs by 70. Birds only begin at 46, and double birds at 38; they also become very rude before the end of the period. The squares begin at 37; but at 67 notched borders appear, and from 70 to 80 line borders.

Journal of the Anthropological Institute (N.S.), Vol. II, Plate XXXI.

W.M.F. Petrie. Sequences in prehistoric remains. *Journal of the Anthropological Institute of Great Britain and Ireland*, 1899.

Robinson(ian) similarity matrix

W.S. Robinson (1951): *Order n objects (graves), given by their pairwise similarities, in such a way that similar objects (graves) are placed close to each other in the ordering.*

Robinson(ian) similarity matrix

W.S. Robinson (1951): *Order n objects (graves), given by their pairwise similarities, in such a way that similar objects (graves) are placed close to each other in the ordering.*

$$\begin{array}{c|ccccc} & G1 & G2 & G3 & G4 & G5 \\ \hline G1 & 3 & 1 & 2 & 0 & 1 \\ G2 & 1 & 4 & 2 & 3 & 3 \\ G3 & 2 & 2 & 4 & 0 & 2 \\ G4 & 0 & 3 & 0 & 4 & 2 \\ G5 & 1 & 3 & 2 & 2 & 3 \end{array}$$

Robinsonian matrix
A

$$\begin{array}{c|ccccc} & G1 & G3 & G5 & G2 & G4 \\ \hline G1 & 3 & 2 & 1 & 1 & 0 \\ G3 & 2 & 4 & 2 & 2 & 0 \\ G5 & 1 & 2 & 3 & 3 & 2 \\ G2 & 1 & 2 & 3 & 4 & 3 \\ G4 & 0 & 0 & 2 & 3 & 4 \end{array}$$

Robinson matrix
$\Pi A \Pi^\mathsf{T}$

Robinson(ian) similarity matrix

W.S. Robinson (1951): *Order n objects (graves), given by their pairwise similarities, in such a way that similar objects (graves) are placed close to each other in the ordering.*

$$\begin{array}{c} \\ G1 \\ G2 \\ G3 \\ G4 \\ G5 \end{array} \begin{array}{c} \begin{array}{ccccc} G1 & G2 & G3 & G4 & G5 \end{array} \\ \begin{pmatrix} 3 & 1 & 2 & 0 & 1 \\ 1 & 4 & 2 & 3 & 3 \\ 2 & 2 & 4 & 0 & 2 \\ 0 & 3 & 0 & 4 & 2 \\ 1 & 3 & 2 & 2 & 3 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ G1 \\ G3 \\ G5 \\ G2 \\ G4 \end{array} \begin{array}{c} \begin{array}{ccccc} G1 & G3 & G5 & G2 & G4 \end{array} \\ \begin{pmatrix} 3 & 2 & 1 & 1 & 0 \\ 2 & 4 & 2 & 2 & 0 \\ 1 & 2 & 3 & 3 & 2 \\ 1 & 2 & 3 & 4 & 3 \\ 0 & 0 & 2 & 3 & 4 \end{pmatrix} \end{array}$$

Robinsonian matrix A | Robinson matrix $\Pi A \Pi^\mathsf{T}$

Theorem (Kendall 1971)

- *P is Petrie $\iff$ PP^T is Robinson.*

Robinson(ian) similarity matrix

W.S. Robinson (1951): *Order n objects (graves), given by their pairwise similarities, in such a way that similar objects (graves) are placed close to each other in the ordering.*

$$\begin{array}{c} \\ G1\\ G2\\ G3\\ G4\\ G5 \end{array}\begin{array}{c} \begin{array}{ccccc} G1 & G2 & G3 & G4 & G5 \end{array}\\ \begin{pmatrix} 3 & 1 & 2 & 0 & 1\\ 1 & 4 & 2 & 3 & 3\\ 2 & 2 & 4 & 0 & 2\\ 0 & 3 & 0 & 4 & 2\\ 1 & 3 & 2 & 2 & 3 \end{pmatrix}\end{array} \qquad \begin{array}{c} \\ G1\\ G3\\ G5\\ G2\\ G4 \end{array}\begin{array}{c} \begin{array}{ccccc} G1 & G3 & G5 & G2 & G4 \end{array}\\ \begin{pmatrix} 3 & 2 & 1 & 1 & 0\\ 2 & 4 & 2 & 2 & 0\\ 1 & 2 & 3 & 3 & 2\\ 1 & 2 & 3 & 4 & 3\\ 0 & 0 & 2 & 3 & 4 \end{pmatrix}\end{array}$$

Robinsonian matrix
A

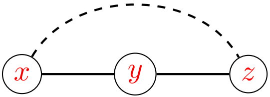
Robinson matrix
$\Pi A \Pi^{\mathsf{T}}$

Theorem (Kendall 1971)

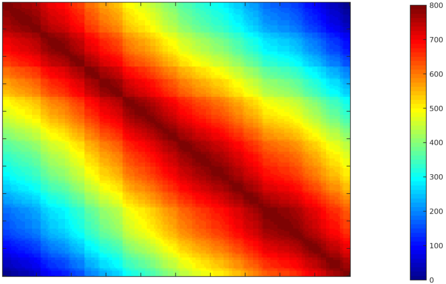
- *P is Petrie $\iff$ PP^T is Robinson.*
- *P has unimodal columns $\iff$ $P \circ P^{\mathsf{T}} = (\sum_z \min\{P_{xz}, P_{yz}\})_{x,y}$ is Robinson.*

Robinson(ian) similarity matrix

$A \in \mathcal{S}^n$ is a **Robinson similarity** if its entries **increase** monotonically along rows and columns when moving toward the diagonal:

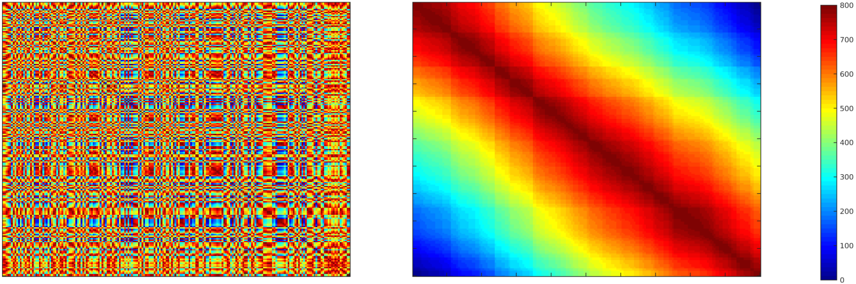


$$A_{xz} \leq \min\{A_{xy}, A_{yz}\}$$
$$\forall\, 1 \leq x < y < z \leq n$$



Robinson(ian) similarity matrix

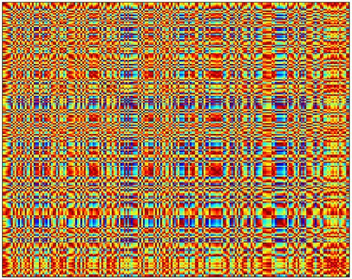
$A \in \mathcal{S}^n$ is a **Robinson similarity** if its entries **increase** monotonically along rows and columns when moving toward the diagonal:

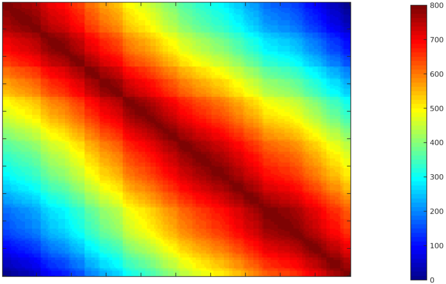


$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\quad \Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

Robinson(ian) similarity matrix

$A \in \mathcal{S}^n$ is a **Robinson similarity** if its entries **increase** monotonically along rows and columns when moving toward the diagonal:



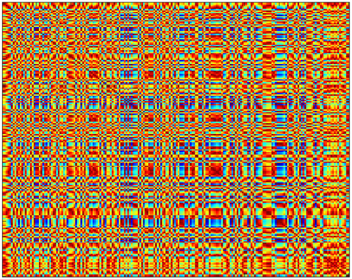


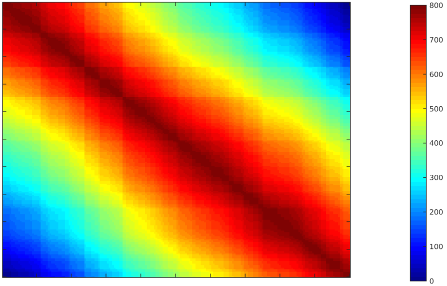
$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\quad \Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

Then π is called a **Robinson ordering** of A.

Robinson(ian) similarity matrix

$A \in \mathcal{S}^n$ is a **Robinson similarity** if its entries **increase** monotonically along rows and columns when moving toward the diagonal:





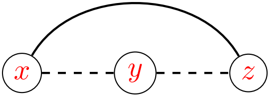
$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

Then π is called a **Robinson ordering** of A.

The **seriation** problem: Find such a **Robinson ordering** π (if it exists).

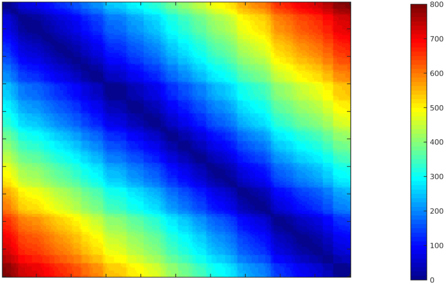
Robinson(ian) dissimilarity matrix

$D \in \mathcal{S}^n$ is a **Robinson** **dissimilarity** if its entries **decrease** monotonically along rows and columns when moving toward the diagonal:



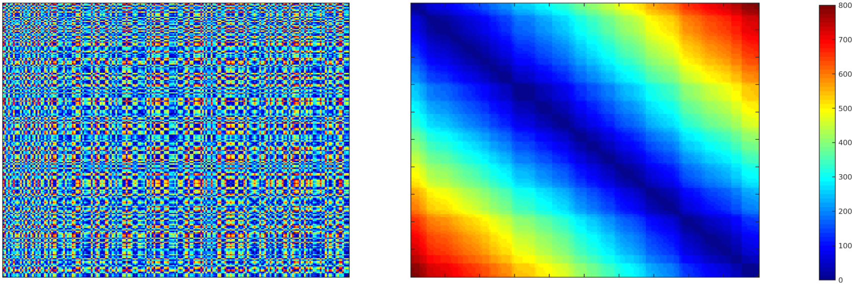
$$D_{xz} \geq \max\{D_{xy}, D_{yz}\}$$

$$\forall\, 1 \leq x < y < z \leq n$$



Robinson(ian) dissimilarity matrix

$D \in \mathcal{S}^n$ is a **Robinson dissimilarity** if its entries **decrease** monotonically along rows and columns when moving toward the diagonal:



$D \in \mathcal{S}^n$ is a **Robinsonian dissimilarity** if there exists a permutation π such that $D^{\pi} := \left(D_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson dissimilarity**,

that is: $A = -D$ is a Robinsonian similarity.

The seriation problem

*Given $A \in \mathcal{S}^n$, find a permutation π (***Robinson ordering***) for which A^π is Robinson or decide that none exists.*

Applications: archeology, ecology, biology (DNA sequencing), ranking, combinatorial data analysis, etc.

The seriation problem

*Given $A \in \mathcal{S}^n$, find a permutation π (**Robinson ordering**) for which A^π is Robinson or decide that none exists.*

Applications: archeology, ecology, biology (DNA sequencing), ranking, combinatorial data analysis, etc.

Optimization approach via Quadratic Assignment:

$$\mathrm{QAP}(A, D) \qquad \min_\pi \sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)}.$$

- With $D = ((x-y)^2)$

The seriation problem

*Given $A \in \mathcal{S}^n$, find a permutation π (**Robinson ordering**) for which A^π is Robinson or decide that none exists.*

Applications: archeology, ecology, biology (DNA sequencing), ranking, combinatorial data analysis, etc.

Optimization approach via Quadratic Assignment:

$$\mathrm{QAP}(A, D) \qquad \min_{\pi} \sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)}.$$

- With $D = ((x-y)^2) \rightsquigarrow$ 2-SUM problem, NP-hard for general A [George-Pothen 97]

The seriation problem

*Given $A \in \mathcal{S}^n$, find a permutation π (**Robinson ordering**) for which A^π is Robinson or decide that none exists.*

Applications: archeology, ecology, biology (DNA sequencing), ranking, combinatorial data analysis, etc.

Optimization approach via Quadratic Assignment:

$$\mathrm{QAP}(A, D) \qquad \min_{\pi} \sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)}.$$

- With $D = ((x-y)^2) \rightsquigarrow$ 2-SUM problem, NP-hard for general A [George-Pothen 97]
- Motivates the spectral algorithm of [Atkins-Boman-Hendrickson 98]

The seriation problem

*Given $A \in \mathcal{S}^n$, find a permutation π (**Robinson ordering**) for which A^π is Robinson or decide that none exists.*

Applications: archeology, ecology, biology (DNA sequencing), ranking, combinatorial data analysis, etc.

Optimization approach via Quadratic Assignment:

$$\mathrm{QAP}(A,D) \qquad\qquad \min_{\pi} \sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)}.$$

- With $D = ((x-y)^2) \rightsquigarrow$ 2-SUM problem, NP-hard for general A [George-Pothen 97]
- Motivates the spectral algorithm of [Atkins-Boman-Hendrickson 98]
- Note D is a Robinson dissimilarity & Toeplitz
  $\rightsquigarrow$ QAP(A,D) is poly-time solvable if A is a Robinsonian similarity

An easy instance of QAP

Theorem (L-Seminaroti 2015)

1. *If A is a Robinson similarity, D is a Robinson dissimilarity, and A or D is Toeplitz, then the identity permutation solves QAP(A, D) at optimality.*

An easy instance of QAP

Theorem (L-Seminaroti 2015)

1. *If A is a Robinson similarity, D is a Robinson dissimilarity, and A or D is Toeplitz, then the identity permutation solves QAP(A, D) at optimality.*
2. *If π is a Robinson (similarity) ordering of A, σ is a Robinson (dissimilarity) ordering of D, and A^π or D^σ is Toeplitz, then $\sigma^{-1}\pi$ solves QAP(A, D) at optimality.*

An easy instance of QAP

Theorem (L-Seminaroti 2015)

1. *If A is a Robinson similarity, D is a Robinson dissimilarity, and A or D is Toeplitz, then the identity permutation solves QAP(A, D) at optimality.*
2. *If π is a Robinson (similarity) ordering of A, σ is a Robinson (dissimilarity) ordering of D, and A^π or D^σ is Toeplitz, then $\sigma^{-1}\pi$ solves QAP(A, D) at optimality.*

Contains the special case when A is a block matrix:

0	0	0
0	1	0
0	0	0

and $D = ((x-y)^2)$

[Fogel-Jenatton-Bach-Aspremont NIPS'13]

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v$

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v$ s.t. $\|v\| = 1, \ e^\mathsf{T} v = 0.$

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v \quad \text{s.t.} \quad \|v\| = 1, \; e^\mathsf{T} v = 0.$$

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v \quad \text{s.t.} \quad \|v\| = 1, \ e^\mathsf{T} v = 0.$$

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then its Laplacian matrix $L_A := \mathit{Diag}(Ae) - A$ has a* **monotone** *Fiedler vector.*

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v \quad \text{s.t.} \quad \|v\| = 1, \ e^\mathsf{T} v = 0.$$

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then its Laplacian matrix $L_A := \text{Diag}(Ae) - A$ has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$. If A is Robinsonian then $\lambda_2(L_A) > \mathbf{0}$ and $\lambda_2(L_A)$ is* **simple**.

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v \quad \text{s.t.} \quad \|v\| = 1, \; e^\mathsf{T} v = 0.$$

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then its Laplacian matrix $L_A := \mathit{Diag}(Ae) - A$ has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$. If A is Robinsonian then $\lambda_2(L_A) > \mathbf{0}$ and $\lambda_2(L_A)$ is* **simple**.
3. *If* **the** *Fiedler vector v_2 has* **no repeated entries**, *then a permutation π orders v_2 monotonically $\iff \pi$ is a Robinson ordering of A.*

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v \quad \text{s.t.} \quad \|v\| = 1, \; e^\mathsf{T} v = 0.$$

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then its Laplacian matrix $L_A := \text{Diag}(Ae) - A$ has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$. If A is Robinsonian then $\lambda_2(L_A) > \mathbf{0}$ and $\lambda_2(L_A)$ is* **simple**.
3. *If* **the** *Fiedler vector v_2 has* **no repeated entries**, *then a permutation π orders v_2 monotonically $\iff$ π is a Robinson ordering of A.*

General case: If v_2 has repeated entries, then **recurse** the algorithm on the submatrices indexed by the repeated entries.

The spectral algorithm to recognize Robinsonian matrices

Given $A \geq 0$: "Relax" 2-SUM: $\min_\pi \sum_{x,y} A_{xy}(\pi(x) - \pi(y))^2$ by

$\min_{v \in \mathbb{R}^n} \sum_{x,y} A_{xy}(v_x - v_y)^2 = v^\mathsf{T} L_A v$ s.t. $\|v\| = 1$, $e^\mathsf{T} v = 0$.

$\rightsquigarrow$ **Fiedler value**: $\lambda_2(L_A)$, whose eigenvectors are the **Fiedler vectors**.

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then its Laplacian matrix $L_A := \text{Diag}(Ae) - A$ has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$. If A is Robinsonian then $\lambda_2(L_A) >$* $\mathbf{0}$ *and $\lambda_2(L_A)$ is* **simple**.
3. *If* **the** *Fiedler vector v_2 has* **no repeated entries**, *then a permutation π orders v_2 monotonically $\Longleftrightarrow$ π is a Robinson ordering of A.*

General case: If v_2 has repeated entries, then **recurse** the algorithm on the submatrices indexed by the repeated entries.

Can encode **all** Robinson orderings of A using PQ-trees.

Combinatorial algorithms

Interval (hyper)graphs

Unit interval graphs

Links to interval (hyper)graphs

For a similarity $A \in \mathcal{S}^n$, a **ball** is any set $B(x, \delta) = \{y \in [n], A_{xy} \geq \delta\}$.

$\mathcal{B}$: set of all balls; $V = [n]$.

Theorem (Mirkin-Rodin 1984)

The following are equivalent:

1. *A is a Robinsonian similarity*
2. *the ball hypergraph $\mathcal{H} = (V, \mathcal{B})$ is an* **interval hypergraph***: its vertices/hyperedges incidence matrix has C1P*

Links to interval (hyper)graphs

For a similarity $A \in \mathcal{S}^n$, a **ball** is any set $B(x, \delta) = \{y \in [n], A_{xy} \geq \delta\}$.

$\mathcal{B}$: set of all balls; $V = [n]$.

Theorem (Mirkin-Rodin 1984)

The following are equivalent:

1. *A is a Robinsonian similarity*
2. *the ball hypergraph $\mathcal{H} = (V, \mathcal{B})$ is an* **interval hypergraph***:*
   its vertices/hyperedges incidence matrix has C1P
3. *the intersection graph of $\mathcal{B}$ is an* **interval graph** $\iff$
   its max.cliques/vertices incidence matrix has C1P [Fulkerson-Gross 65]

Links to interval (hyper)graphs

For a similarity $A \in \mathcal{S}^n$, a **ball** is any set $B(x, \delta) = \{y \in [n], A_{xy} \geq \delta\}$.

$\mathcal{B}$: set of all balls; $V = [n]$.

Theorem (Mirkin-Rodin 1984)

The following are equivalent:

1. *A is a Robinsonian similarity*
2. *the ball hypergraph $\mathcal{H} = (V, \mathcal{B})$ is an* ***interval hypergraph****:*
   its vertices/hyperedges incidence matrix has C1P
3. *the intersection graph of $\mathcal{B}$ is an* ***interval graph*** $\Longleftrightarrow$
   its max.cliques/vertices incidence matrix has C1P [Fulkerson-Gross 65]

Can test whether $M \in \{0, 1\}^{p \times q}$ with m ones has C1P in $O(p + q + m)$ using PQ-trees. [Booth-Lueker 76]

Existing recognition algorithms for Robinsonian matrices

	Year	Complexity	Subroutine	Paradigm
Mirkin & Rodin	1984	$O(n^4)$	PQ-trees	interval hypergraphs
Chepoi & Fichet	1997	$O(n^3)$	PQ-trees	interval hypergraphs
Préa & Fortin	2014	$O(n^2)$	PQ-trees	interval graphs
Atkins et al.	1998	$O(n(T(n) + n \log n))$	eigenvalues	Fiedler vector
Laurent & Seminaroti	2015	$O(L(m + n))$	Lex-BFS	unit interval graphs
Laurent & Seminaroti	2017	$O(n^2 + mn \log n)$	SFS	new weighted graph search

n: size of A; m : # of nonzero entries of A; L : # of distinct values of A.

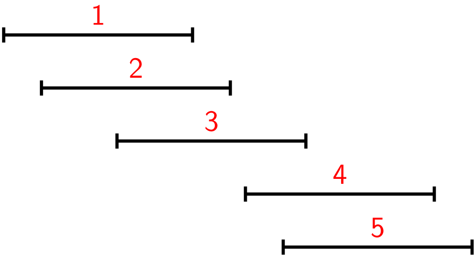
Binary Robinsonian matrices and unit interval graphs

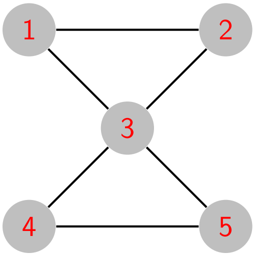
Fact (Roberts 1969)

$A \in \{0,1\}^{n\times n}$ is a Robinsonian similarity if and only if A is the adjacency matrix of a **unit interval graph** *G.*

G is a **unit interval graph** if $\exists$ unit intervals $I_1, \ldots, I_n$ in $\mathbb{R}$ such that

$$\{x,y\} \in E \quad \Longleftrightarrow \quad I_x \cap I_y \neq \emptyset.$$





Binary Robinsonian matrices and unit interval graphs

Fact (Roberts 1969)

$A \in \{0,1\}^{n\times n}$ *is a Robinsonian similarity if and only if* A *is the adjacency matrix of a* **unit interval graph** G.

Theorem (Looges-Olariu 1993)

G *is a* **unit interval graph** *if and only if there exists a linear order* π *of the vertices satisfying the* **3-point condition***:*

$$\{x,z\} \in E \quad \Longrightarrow \quad \{x,y\},\{y,z\} \in E \quad \text{if} \;\; x <_\pi y <_\pi z$$

Binary Robinsonian matrices and unit interval graphs

Fact (Roberts 1969)

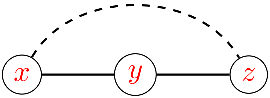
$A \in \{0,1\}^{n\times n}$ is a Robinsonian similarity if and only if A is the adjacency matrix of a **unit interval graph** *G.*

Theorem (Looges-Olariu 1993)

G is a **unit interval graph** *if and only if there exists a linear order π of the vertices satisfying the* **3-point condition***:*

$$\{x,z\} \in E \quad \Longrightarrow \quad \{x,y\},\{y,z\} \in E \quad \text{if} \;\; x <_\pi y <_\pi z$$

Recall the Robinson (similarity) property:



$$A_{xz} \leq \min\{A_{xy}, A_{yz}\} \quad \text{if} \;\; x < y < z$$

Binary Robinsonian matrices and unit interval graphs

Fact (Roberts 1969)

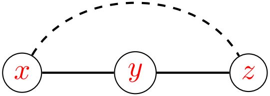
$A \in \{0,1\}^{n\times n}$ is a Robinsonian similarity if and only if A is the adjacency matrix of a **unit interval graph** *G.*

Theorem (Looges-Olariu 1993)

G is a **unit interval graph** *if and only if there exists a linear order π of the vertices satisfying the* **3-point condition***:*

$$\{x,z\} \in E \quad \Longrightarrow \quad \{x,y\},\{y,z\} \in E \quad \text{if} \;\; x <_\pi y <_\pi z$$

Recall the Robinson (similarity) property:



$$A_{xz} \le \min\{A_{xy}, A_{yz}\} \quad \text{if} \;\; x < y < z$$

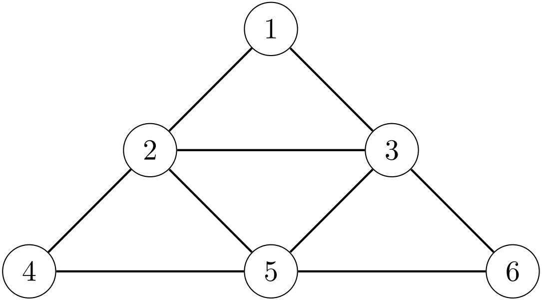
Theorem (Corneil 2004)

One can recognize unit interval graphs in $O(|V|+|E|)$ using Lex-BFS.

Graph search: Lex-BFS

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

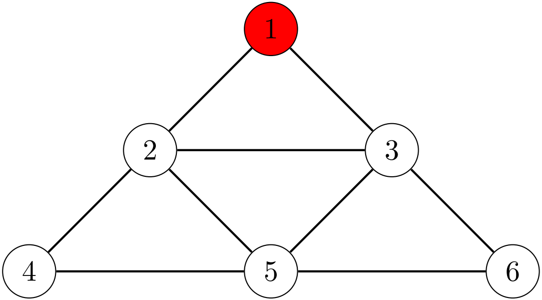
pivot

Q :

1	2	3	4	5	6

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

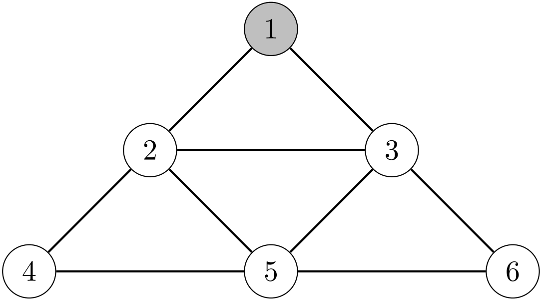
pivot

Q :

1	2	3	4	5	6

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

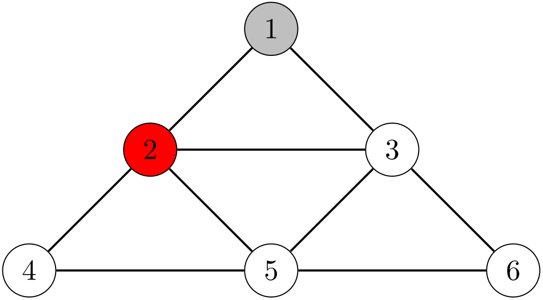
unvisited vertices
(stored in a queue Q)

pivot

$Q:$ | 2 | 3 | 4 | 5 | 6 |

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

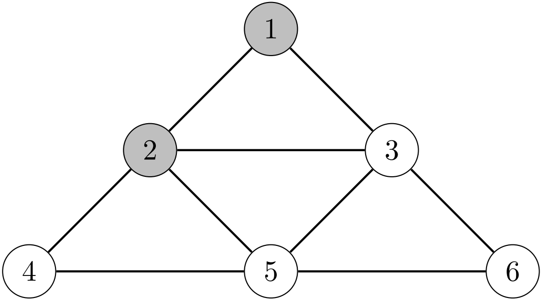
unvisited vertices
(stored in a queue Q)

pivot

$Q:$ | 2 | 3 | 4 | 5 | 6 |

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

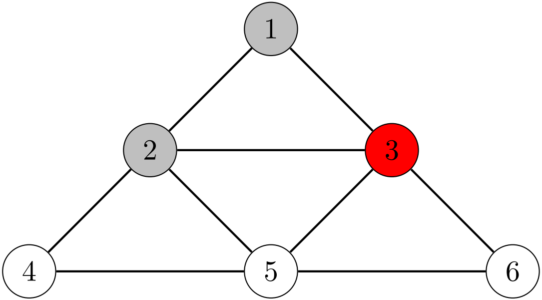
pivot

Q :

3	4	5	6

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

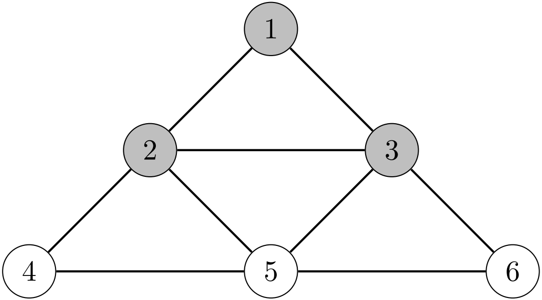
pivot

3	4	5	6

$Q:$

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

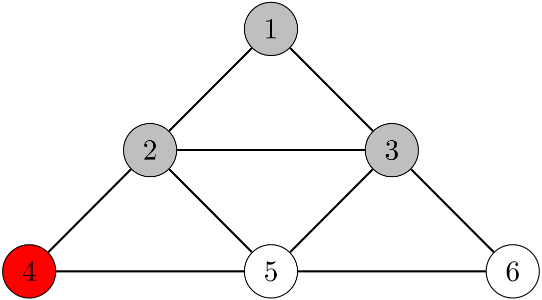
pivot

Q :

4	5	6

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

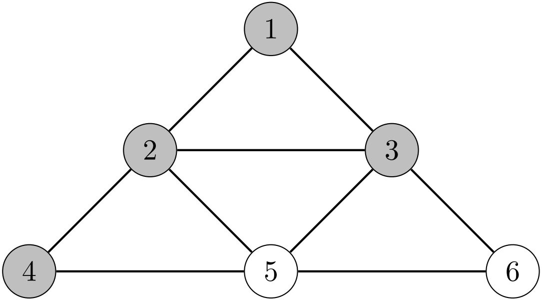
pivot

Q :

4	5	6

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

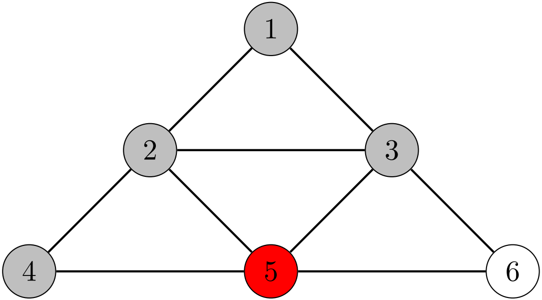
unvisited vertices
(stored in a queue Q)

pivot

$Q:$ | 5 | 6 |

Graph search paradigm

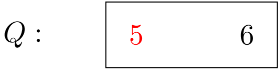
Given a graph $G = (V, E)$:



visited vertices

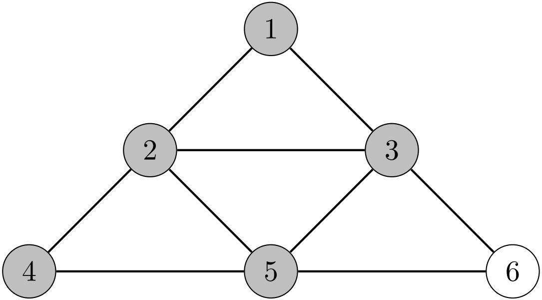
unvisited vertices
(stored in a queue Q)

pivot



Graph search paradigm

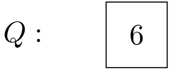
Given a graph $G = (V, E)$:



visited vertices

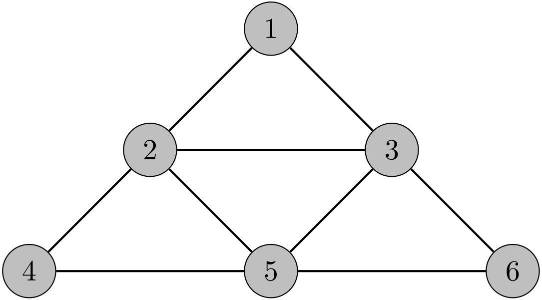
unvisited vertices
(stored in a queue Q)

pivot



Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

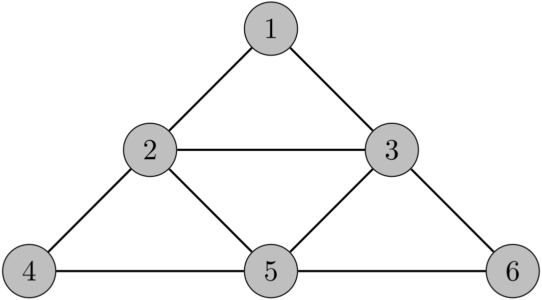
unvisited vertices
(stored in a queue Q)

pivot

$Q:$ $\varnothing$

Graph search paradigm

Given a graph $G = (V, E)$:



visited vertices

unvisited vertices
(stored in a queue Q)

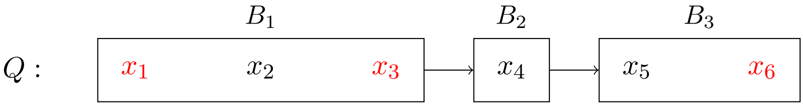
pivot

Different queue updates lead to different graph search algorithms:

- Breadth-First Search (BFS)
- Depth-First Search (DFS)
- Lexicographic Breadth-First Search (Lex-BFS)

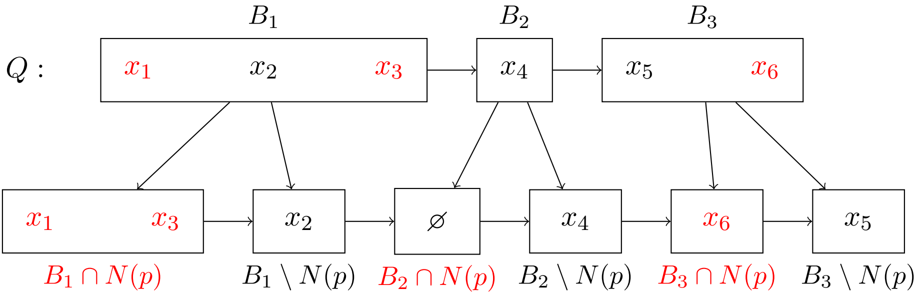
Lex-BFS via partition refinement

Let $N(p)$ denote the neighborhood of the current pivot p.



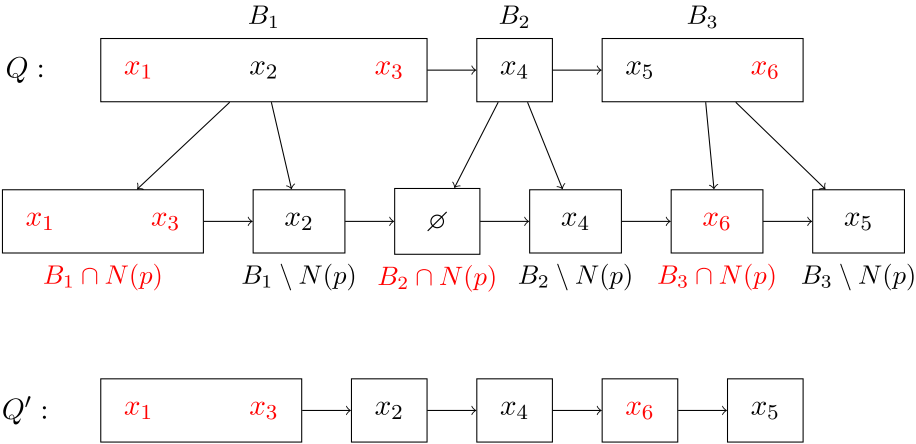
Lex-BFS via partition refinement

Let $N(p)$ denote the neighborhood of the current pivot p.



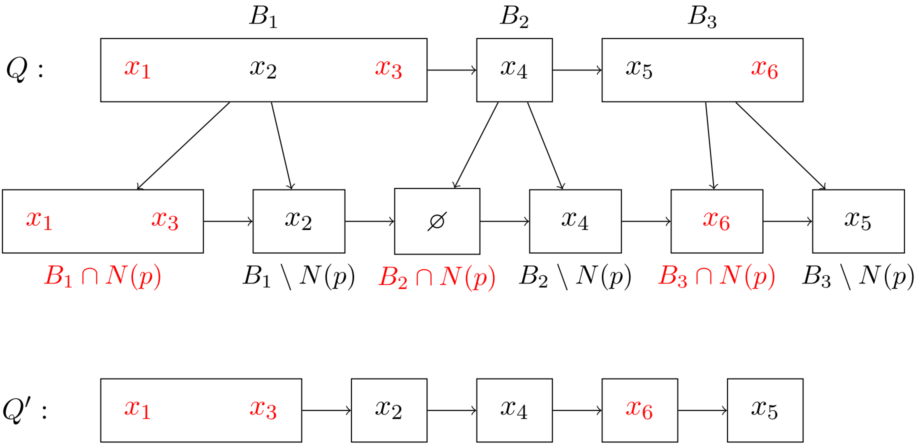
Lex-BFS via partition refinement

Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS via partition refinement

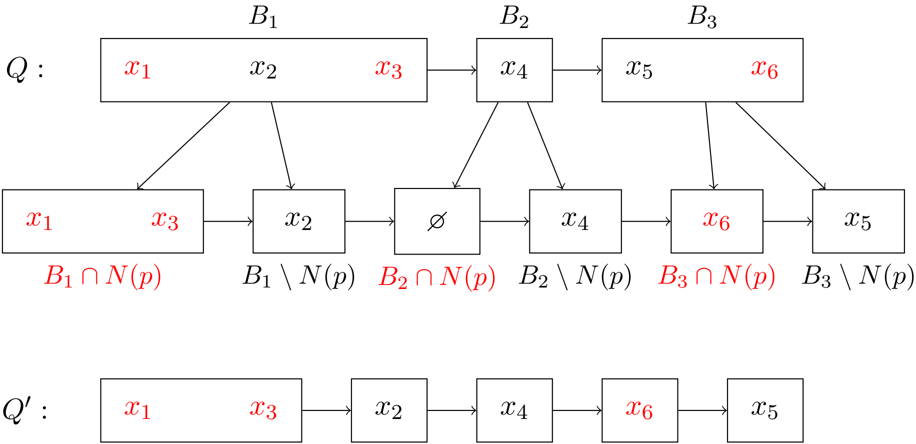
Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS$_+$: Order the vertices in each block according to a given order τ

Lex-BFS via partition refinement

Let $N(p)$ denote the neighborhood of the current pivot p.

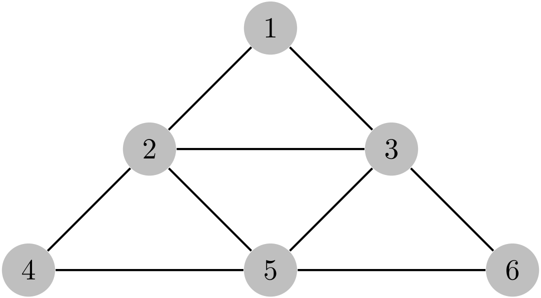


Lex-BFS$_+$: Order the vertices in each block according to a given order τ

Lex-BFS runs in time $O(|V| + |E|)$ [Rose-Tarjan'75, Habib et al.'00]

Example of Lex-BFS$_+$

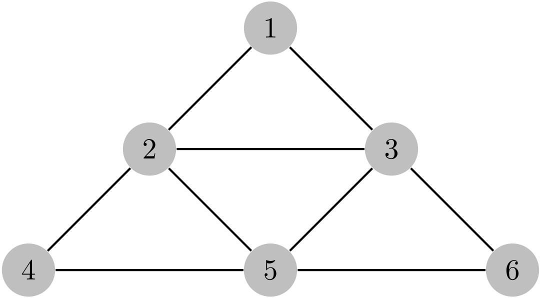
$\tau = (1, 2, 3, 4, 5, 6)$



1	2	3	4	5	6

Example of Lex-BFS$_+$

$\tau = (1, 2, 3, 4, 5, 6)$

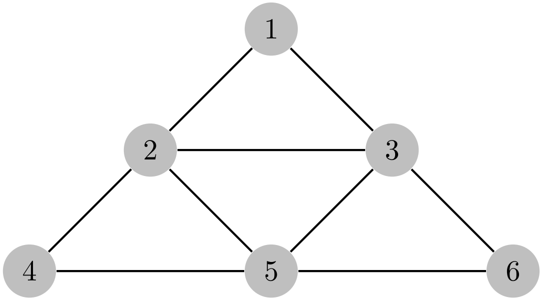


[1 2 3 4 5 6]

1 [2 3] [4 5 6]

Example of Lex-BFS$_+$

$\tau = (1, 2, 3, 4, 5, 6)$



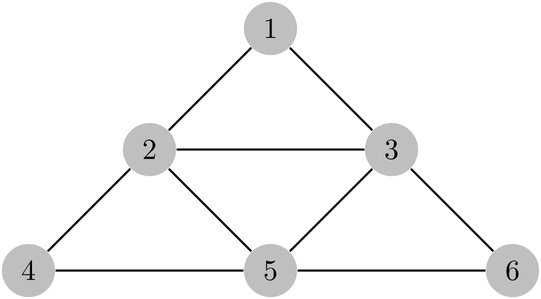
[1 2 3 4 5 6]

1 [2 3] [4 5 6]

1 2 [3] [4 5] [6]

Example of Lex-BFS$_+$

$\tau = (1, 2, 3, 4, 5, 6)$



$\boxed{1\quad 2\quad 3\quad 4\quad 5\quad 6}$ $\qquad$ $1\quad 2\quad \boxed{3}\ \boxed{4\quad 5}\ \boxed{6}$

$1\quad \boxed{2\quad 3}\ \boxed{4\quad 5\quad 6}$ $\qquad$ $1\quad 2\quad 3\quad \boxed{5}\ \boxed{4}\ \boxed{6}$

The Lex-BFS$_+$ ordering is $\sigma = (1, 2, 3, 5, 4, 6)$

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ = \text{Lex-BFS}_+(G, \sigma^{-1})$

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π
5. **else return** "G is not a unit interval graph"

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π
5. **else return** "G is not a unit interval graph"

Hence: In time $O(|V| + |E|)$, return a Robinson ordering of A_G or state A_G is not Robinsonian.

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π
5. **else return** "G is not a unit interval graph"

Hence: In time $O(|V| + |E|)$, return a Robinson ordering of A_G or state A_G is not Robinsonian.

What about general matrices A?

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π
5. **else return** "G is not a unit interval graph"

Hence: In time $O(|V| + |E|)$, return a Robinson ordering of A_G or state A_G is not Robinsonian.

What about general matrices A?

Option 1: Use Lex-BFS for the 'level graphs' of A. [L-Seminaroti'15]

Corneil (2004) 3-sweep algorithm for unit interval graphs

Input: A graph $G = (V, E)$.
Output: an ordering π of V satisfying the 3-point condition, or stating that G is not a unit interval graph.

1. $\sigma =$ Lex-BFS (G)
2. $\sigma_+ =$ Lex-BFS$_+(G, \sigma^{-1})$
3. $\pi =$ Lex-BFS$_+(G, \sigma_+^{-1})$
4. **if** π satisfies 3-vertex condition **return** π
5. **else return** "G is not a unit interval graph"

Hence: In time $O(|V| + |E|)$, return a Robinson ordering of A_G or state A_G is not Robinsonian.

What about general matrices A?

Option 1: Use Lex-BFS for the 'level graphs' of A. [L-Seminaroti'15]

Option 2: Generalize Lex-BFS to weighted graphs: SFS.

Recognizing Robinsonian matrices with Lex-BFS

Lemma

Consider $A \in \mathcal{S}^n$ taking values $\alpha_0 = 0 < \alpha_1 < \alpha_2 < \ldots < \alpha_L$.

A is Robinson $\Longleftrightarrow$ A is a conic combination of $0/1$ Robinson matrices:

$$A = \sum_{l=1}^{L} (\alpha_l - \alpha_{l-1}) A_{G_l},$$

where graph G_l has edges $\{x, y\}$ with $A_{xy} \geq \alpha_l$.

Algorithm (rough sketch):

1. Find the level graphs $G_1, \ldots, G_L$ of A.
2. Find an ordering π of V which **satisfies the 3-point condition for all graphs** G_l $(l = 1, \ldots, L)$. Then π is a Robinson ordering of A. If none exists, then A is not Robinsonian.

$\rightsquigarrow$ algorithm in $O(L(n+m))$ [L-Seminaroti 2015]

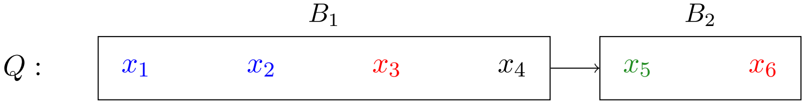
Weighted graph search:

Similarity-First Search (SFS)

Similarity-First Search (SFS) for nonnegative A

For the current pivot p, define $N(p) = \{x : A_{px} > 0\}$.

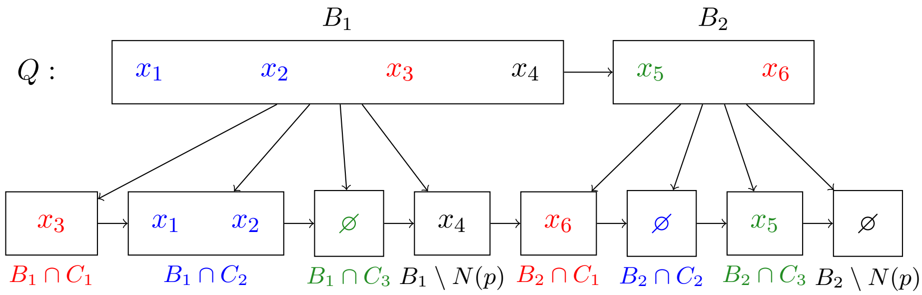
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \ldots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \ldots > 0 \ \ \forall x \in C_1, y \in C_2, z \in C_3, \ldots$



Similarity-First Search (SFS) for nonnegative A

For the current pivot p, define $N(p) = \{x : A_{px} > 0\}$.

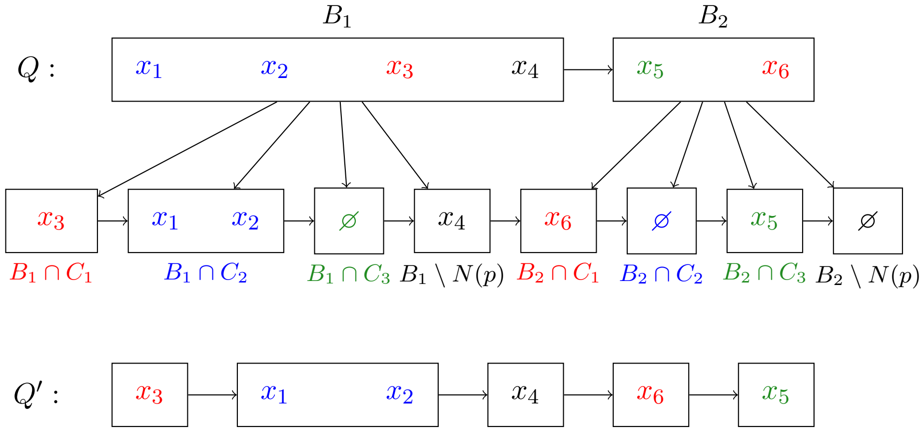
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \ldots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \ldots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \ldots$



Similarity-First Search (SFS) for nonnegative A

For the current pivot p, define $N(p) = \{x : A_{px} > 0\}$.

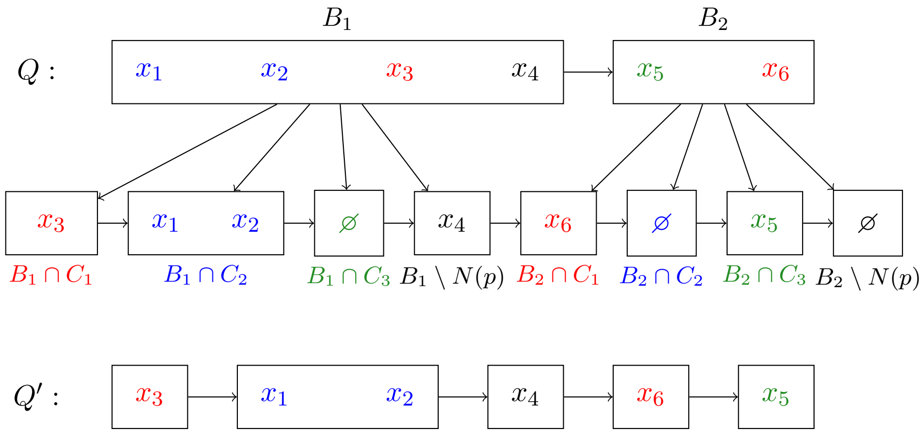
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \dots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \dots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \dots$



Similarity-First Search (SFS) for nonnegative A

For the current pivot p, define $N(p) = \{x : A_{px} > 0\}$.

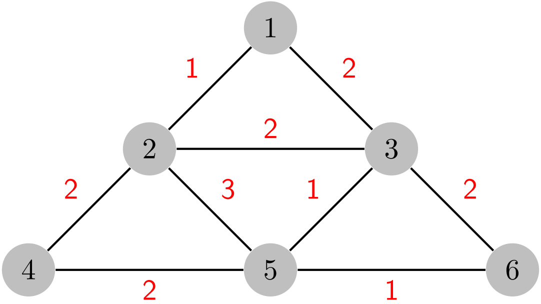
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \dots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \dots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \dots$



SFS runs in $O(n + m \log n)$ if A has m nonzero entries. [L-Seminaroti 17]

Example for SFS_+

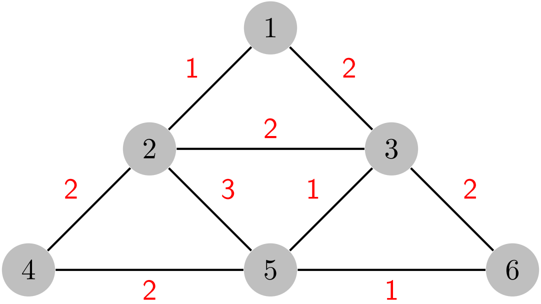
$\tau = (1, 2, 3, 4, 5, 6)$



1	2	3	4	5	6

Example for SFS_+

$\tau = (1, 2, 3, 4, 5, 6)$

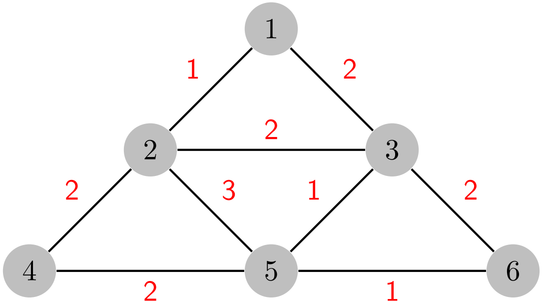


[1 2 3 4 5 6]

1 [3] [2] [4 5 6]

Example for SFS_+

$\tau = (1, 2, 3, 4, 5, 6)$



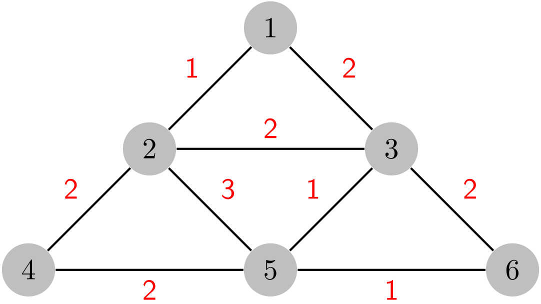
| 1 2 3 4 5 6 |

1 | 3 | 2 | 4 5 6 |

1 3 | 2 | 6 | 5 | 4 |

Example for SFS_+

$\tau = (1, 2, 3, 4, 5, 6)$



1 2 3 4 5 6

1 3 2 4 5 6

1 3 2 6 5 4

1 3 2 6 5 4

The SFS_+ ordering is $\sigma = (1, 3, 2, 6, 5, 4)$

SFS and Robinson matrices

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$

Output: a Robinson ordering π of A, or stating that A is not Robinsonian

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 = \text{SFS}\ (A)$

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$

Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 =$ SFS (A)
2. **for** $i = 1, \ldots, n-2$

5. **end**

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 = \mathsf{SFS}\ (A)$
2. **for** $i = 1, \ldots, n-2$
3. $\quad \sigma_i = \mathsf{SFS}_+(A, \sigma_{i-1}^{-1})$

5. **end**

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 =$ SFS (A)
2. **for** $i = 1, \dots, n-2$
3. $\quad \sigma_i = \text{SFS}_+(A, \sigma_{i-1}^{-1})$
4. $\quad$ if σ_i is a Robinson ordering **return** $\pi = \sigma_i$
5. **end**

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 =$ SFS (A)
2. **for** $i = 1, \ldots, n-2$
3. $\quad \sigma_i =$ SFS$_+(A, \sigma_{i-1}^{-1})$
4. $\quad$ if σ_i is a Robinson ordering **return** $\pi = \sigma_i$
5. **end**
6. **return** "A is not Robinsonian"

Theorem (L-Seminaroti 2017)

Let $A \in \mathcal{S}^n$ be nonnegative with m nonzero entries. Then:

1. *$A \in \mathcal{S}^n$ is Robinsonian $\iff$ σ_{n-2} is a Robinson ordering.*

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 =$ SFS (A)
2. **for** $i = 1, \ldots, n-2$
3. $\quad \sigma_i =$ SFS$_+(A, \sigma_{i-1}^{-1})$
4. $\quad$ if σ_i is a Robinson ordering **return** $\pi = \sigma_i$
5. **end**
6. **return** "A is not Robinsonian"

Theorem (L-Seminaroti 2017)

Let $A \in \mathcal{S}^n$ be nonnegative with m nonzero entries. Then:

1. *$A \in \mathcal{S}^n$ is Robinsonian $\iff$ σ_{n-2} is a Robinson ordering.*
2. *The multisweep recognition algorithm runs in $O(n^2 + mn \log n)$ time.*

SFS multisweep recognition algorithm

Input: a nonnegative matrix $A \in \mathcal{S}^n$
Output: a Robinson ordering π of A, or stating that A is not Robinsonian

1. $\sigma_0 =$ SFS (A)
2. **for** $i = 1, \ldots, n-2$
3. $\quad \sigma_i =$ SFS$_+(A, \sigma_{i-1}^{-1})$
4. $\quad$ if σ_i is a Robinson ordering **return** $\pi = \sigma_i$
5. **end**
6. **return** "A is not Robinsonian"

Theorem (L-Seminaroti 2017)

Let $A \in \mathcal{S}^n$ be nonnegative with m nonzero entries. Then:

1. *$A \in \mathcal{S}^n$ is Robinsonian $\iff$ σ_{n-2} is a Robinson ordering.*
2. *The multisweep recognition algorithm runs in $O(n^2 + mn \log n)$ time.*
3. *Simpler test at line 4: Check whether $\sigma_i = \sigma_{i-1}^{-1}$. If* **YES** *then: if σ_i is Robinson then A is Robinsonian; else A is not Robinsonian.*

SFS and end vertices of Robinson orderings (anchors of A)

SFS and end vertices of Robinson orderings (anchors of A)

- $a \in V$ is an **anchor** of A if there exists a Robinson ordering π of A starting (or ending) at a

$$\pi: \quad a \quad a_1 \quad a_2 \quad \cdots \quad b_2 \quad b_1 \quad b$$

SFS and end vertices of Robinson orderings (anchors of A)

- $a \in V$ is an **anchor** of A if there exists a Robinson ordering π of A starting (or ending) at a
- a, $b \in V$ are **opposite anchors** of A if there exists a Robinson ordering π of A starting at a and ending at b

$$\pi : \quad a \quad a_1 \quad a_2 \quad \cdots \quad b_2 \quad b_1 \quad b$$

SFS and end vertices of Robinson orderings (anchors of A)

- $a \in V$ is an **anchor** of A if there exists a Robinson ordering π of A starting (or ending) at a
- a, $b \in V$ are **opposite anchors** of A if there exists a Robinson ordering π of A starting at a and ending at b

$$\sigma: \quad a \quad a_1 \quad a_2 \quad \cdots \quad b_2 \quad b_1 \quad b$$

Theorem (L-Seminaroti 2017)

Assume A is Robinsonian and $\sigma = SFS(A)$ has **last vertex** *b.*

1. *b is an anchor of A.*
   (In fact any anchor arises as end vertex of some SFS ordering of A.)

SFS and end vertices of Robinson orderings (anchors of A)

- $a \in V$ is an **anchor** of A if there exists a Robinson ordering π of A starting (or ending) at a
- a, $b \in V$ are **opposite anchors** of A if there exists a Robinson ordering π of A starting at a and ending at b

$$\sigma : \quad a \quad a_1 \quad a_2 \quad \cdots \quad b_2 \quad b_1 \quad b$$

Theorem (L-Seminaroti 2017)

Assume A is Robinsonian and $\sigma = SFS(A)$ has **last vertex** *b.*

1. *b is an anchor of A.*
   (In fact any anchor arises as end vertex of some SFS ordering of A.)
2. *If the* **first vertex** *a in σ is an anchor of A, then a, b are opposite anchors of A.*

SFS and end vertices of Robinson orderings (anchors of A)

- $a \in V$ is an **anchor** of A if there exists a Robinson ordering π of A starting (or ending) at a
- a, $b \in V$ are **opposite anchors** of A if there exists a Robinson ordering π of A starting at a and ending at b

$$\sigma : \quad a \quad a_1 \quad a_2 \quad \cdots \quad b_2 \quad b_1 \quad b$$

Theorem (L-Seminaroti 2017)

Assume A is Robinsonian and $\sigma = SFS(A)$ has ***last vertex*** *b.*

1. *b is an anchor of A.*
   (In fact any anchor arises as end vertex of some SFS ordering of A.)
2. *If the* ***first vertex*** *a in σ is an anchor of A, then a, b are opposite anchors of A.*

Key ingredient: combinatorial characterization of (opposite) anchors of A in terms of certain "forbidden paths".

Anchor flipping property of SFS_+

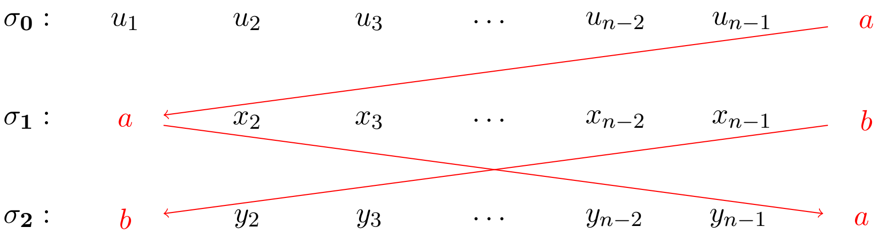
Anchor flipping property of SFS_+

$\boldsymbol{\sigma_0}:$ u_1 u_2 u_3 $\cdots$ u_{n-2} u_{n-1} a

Anchor flipping property of SFS_+

$\boldsymbol{\sigma_0}$:	u_1	u_2	u_3	$\cdots$	u_{n-2}	u_{n-1}	a
$\boldsymbol{\sigma_1}$:	a	x_2	x_3	$\cdots$	x_{n-2}	x_{n-1}	b

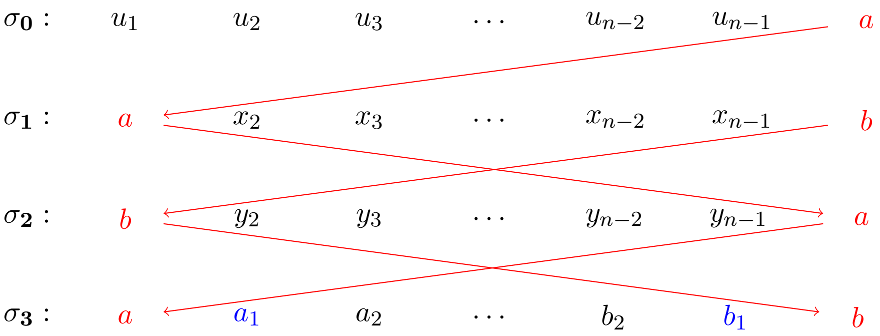
Anchor flipping property of SFS_+



Theorem (Anchors Flipping)

Assume $A \in \mathcal{S}^n$ is Robinsonian and $\sigma_i =$ SFS$_+(A, \sigma_{i-1})$ with $i \geq 1$.
σ_1 start with a and end with b; σ_2 start with b and end with a;

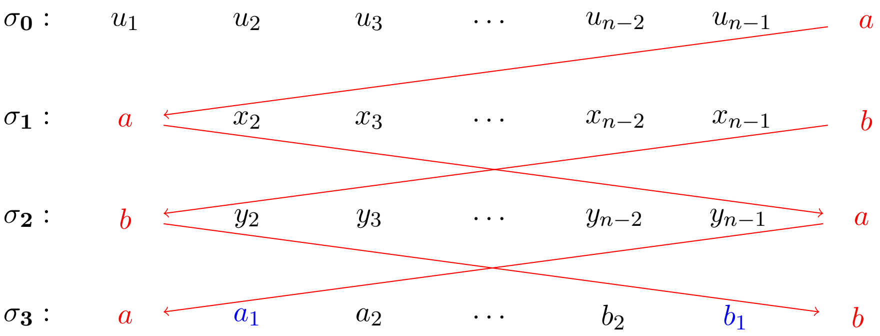
Anchor flipping property of SFS_+



Theorem (Anchors Flipping)

Assume $A \in \mathcal{S}^n$ *is Robinsonian and* $\sigma_i = \textit{SFS}_+(A, \sigma_{i-1})$ *with* $i \geq 1$.
σ_1 , σ_3 *start with* a *and end with* b; σ_2 , σ_4 *start with* b *and end with* a; *etc.*

Anchor flipping property of SFS$_+$



Theorem (Anchors Flipping)

Assume $A \in \mathcal{S}^n$ is Robinsonian and $\sigma_i =$ SFS$_+(A, \sigma_{i-1})$ with $i \geq 1$.
σ_1 , σ_3 start with a and end with b; σ_2 , σ_4 start with b and end with a; etc.

Key fact: $a_1 = y_{n-1}$ and b_1 are opposite anchors of $A[V \setminus \{a, b\}]$.

Anchor flipping property of SFS$_+$

σ_0 :	u_1	u_2	u_3	$\cdots$	u_{n-2}	u_{n-1}	a
σ_1 :	a	x_2	x_3	$\cdots$	x_{n-2}	x_{n-1}	b
σ_2 :	b	y_2	y_3	$\cdots$	y_{n-2}	y_{n-1}	a
σ_3 :	a	a_1	a_2	$\cdots$	b_2	b_1	b

Theorem (Anchors Flipping)

Assume $A \in \mathcal{S}^n$ is Robinsonian and $\sigma_i = SFS_+(A, \sigma_{i-1})$ with $i \geq 1$.
σ_1, σ_3 start with a and end with b; σ_2, σ_4 start with b and end with a; etc.

Moreover: $\sigma_{n-2}[A \setminus \{a, b\}]$ can be seen as result of the multisweep algorithm applied to $A[V \setminus \{a, b\}]$, starting with $\sigma_3[V \setminus \{a, b\}]$.

$\rightsquigarrow$ **can apply induction.**

Crucial technical tool: Path avoiding a vertex

For distinct $x, y, z \in V$, $P = (x = v_0, v_1, \ldots, v_{k-1}, v_k = y)$ is a **path from x to y avoiding z** if each triple (v_i, z, v_{i+1}) is **not Robinson**, i.e.,

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

Crucial technical tool: Path avoiding a vertex

For distinct $x, y, z \in V$, $P = (x = v_0, v_1, \ldots, v_{k-1}, v_k = y)$ is a **path from x to y avoiding z** if each triple (v_i, z, v_{i+1}) is **not Robinson**, i.e.,

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

Fact

Assume A is Robinsonian. If $\exists$ path $x \rightsquigarrow y$ avoiding z then z **does not lie between x and y in any Robinson ordering** π *of A.*

Crucial technical tool: Path avoiding a vertex

For distinct $x, y, z \in V$, $P = (x = v_0, v_1, \ldots, v_{k-1}, v_k = y)$ is a **path from** x **to** y **avoiding** z if each triple (v_i, z, v_{i+1}) is **not Robinson**, i.e.,

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

Fact

Assume A is Robinsonian. If $\exists$ path $x \rightsquigarrow y$ avoiding z then z **does not lie between** x **and** y **in any Robinson ordering** π *of A.*

Theorem

a is an anchor of $A \iff \nexists\, u, v \in V$, a path $a \rightsquigarrow u$ avoiding v, and a path $a \rightsquigarrow v$ avoiding u (since $\pi : a \cdots v \cdots u$ or $\pi : a \cdots u \cdots v$)

Crucial technical tool: Path avoiding a vertex

For distinct $x, y, z \in V$, $P = (x = v_0, v_1, \ldots, v_{k-1}, v_k = y)$ is a **path from x to y avoiding z** if each triple (v_i, z, v_{i+1}) is **not Robinson**, i.e.,

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

Fact

Assume A is Robinsonian. If $\exists$ path $x \rightsquigarrow y$ avoiding z then z ***does not lie between*** *x* ***and*** *y* ***in any Robinson ordering*** *π of A.*

Theorem

a is an anchor of $A \iff \nexists\ u, v \in V$, a path $a \rightsquigarrow u$ avoiding v, and a path $a \rightsquigarrow v$ avoiding u (since $\pi : a \cdots v \cdots u$ or $\pi : a \cdots u \cdots v$)

Theorem

Two anchors a, b of A are ***opposite anchors***
$\iff \nexists$ path $a \rightsquigarrow b$ avoiding some u (since $\pi : a \cdots u \cdots b$)

Certifying non-Robinsonian matrices

Definition

A **weighted asteroidal triple** for A is a triple $\{x, y, z\}$ such that $\exists$ path $x \rightsquigarrow y$ avoiding z; $\exists$ path $x \rightsquigarrow z$ avoiding y; and $\exists$ path $y \rightsquigarrow z$ avoiding x.

If such triple exists then A is not Robinsonian!

Certifying non-Robinsonian matrices

Definition

A **weighted asteroidal triple** for A is a triple $\{x, y, z\}$ such that
$\exists$ path $x \rightsquigarrow y$ avoiding z; $\exists$ path $x \rightsquigarrow z$ avoiding y; and
$\exists$ path $y \rightsquigarrow z$ avoiding x.

If such triple exists then A is not Robinsonian!

Theorem (L-Seminaroti-Tanigawa 2017)

A is Robinsonian $\iff$ there does not exist a weighted asteroidal triple.

Certifying non-Robinsonian matrices

Definition

A **weighted asteroidal triple** for A is a triple $\{x, y, z\}$ such that
$\exists$ path $x \rightsquigarrow y$ avoiding z; $\exists$ path $x \rightsquigarrow z$ avoiding y; and
$\exists$ path $y \rightsquigarrow z$ avoiding x.

If such triple exists then A is not Robinsonian!

Theorem (L-Seminaroti-Tanigawa 2017)

A is Robinsonian $\iff$ there does not exist a weighted asteroidal triple.

- Can find a weighted asteroidal triple in $O(n^3)$:

Certifying non-Robinsonian matrices

Definition

A **weighted asteroidal triple** for A is a triple $\{x, y, z\}$ such that
$\exists$ path $x \rightsquigarrow y$ avoiding z; $\exists$ path $x \rightsquigarrow z$ avoiding y; and
$\exists$ path $y \rightsquigarrow z$ avoiding x.

If such triple exists then A is not Robinsonian!

Theorem (L-Seminaroti-Tanigawa 2017)

A is Robinsonian $\iff$ there does not exist a weighted asteroidal triple.

- Can find a weighted asteroidal triple in $O(n^3)$: this certifies A is **not Robinsonian**.

Certifying non-Robinsonian matrices

Definition

A **weighted asteroidal triple** for A is a triple $\{x, y, z\}$ such that
$\exists$ path $x \rightsquigarrow y$ avoiding z; $\exists$ path $x \rightsquigarrow z$ avoiding y; and
$\exists$ path $y \rightsquigarrow z$ avoiding x.

If such triple exists then A is not Robinsonian!

Theorem (L-Seminaroti-Tanigawa 2017)

A is Robinsonian $\iff$ there does not exist a weighted asteroidal triple.

• Can find a weighted asteroidal triple in $O(n^3)$: this certifies A is **not Robinsonian**.

• This implies the characterization of **unit interval graphs**: no asteroidal triple, no induced cycle of length at least 4, no induced claw $K_{1,3}$

[Roberts 69]

Tight example where $n-1$ sweeps are needed

Example by S. Tanigawa: Robinson matrix $A \in \mathcal{S}^n$:
$A_{1n} = 0,\ A_{1i} = 1,\ A_{2n} = 1,\ A_{in} = 2,\ A_{ij} = A_{i-1,j+1} + 1.$

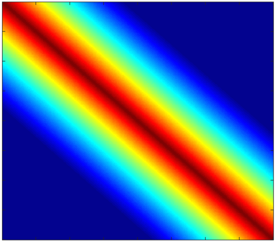
$$A = \begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{5} \\ \mathbf{6} \\ \mathbf{7} \\ \mathbf{8} \\ \mathbf{9} \\ \mathbf{10} \\ \mathbf{11} \end{array} \begin{array}{c} \begin{array}{ccccccccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} & \mathbf{10} & \mathbf{11} \end{array} \\ \begin{pmatrix} * & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ & * & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 \\ & & * & 3 & 3 & 3 & 3 & 3 & 2 & 2 & 2 \\ & & & * & 4 & 4 & 4 & 3 & 3 & 3 & 2 \\ & & & & * & 5 & 4 & 4 & 4 & 3 & 2 \\ & & & & & * & 5 & 5 & 4 & 3 & 2 \\ & & & & & & * & 5 & 4 & 3 & 2 \\ & & & & & & & * & 4 & 3 & 2 \\ & & & & & & & & * & 3 & 2 \\ & & & & & & & & & * & 2 \\ & & & & & & & & & & * \end{pmatrix} \end{array}$$

With SFS $\sigma_0 = (2, 3, \ldots, n, 1)$, the **first Robinson sweep** is σ_{n-2}.

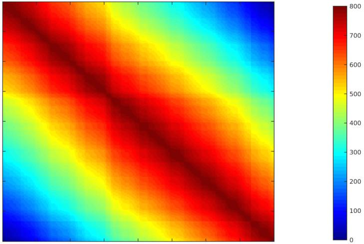
Computational experiments

by Matteo

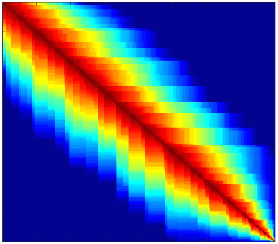
Instances generation



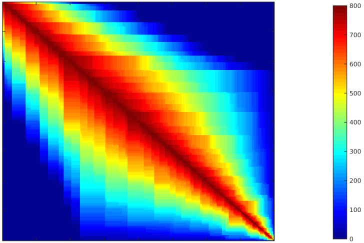
(a) Generation 1



(b) Generation 2



(c) Generation 3



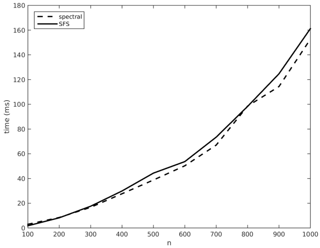
(d) Generation 4

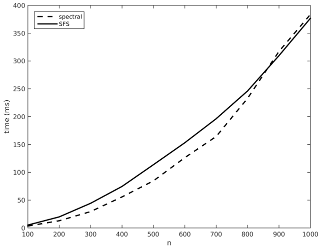
Performance table ($n \leq 1000$)

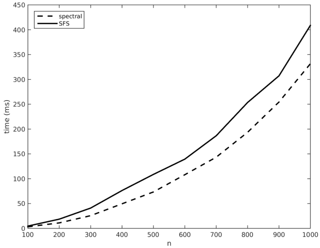
# nonzero entries	# distinct values / algorithms \ n	low (≤ 50) spectral	low SFS	low LBFS	medium (> 50 and ≤ 200) spectral	medium SFS	medium LBFS	high (≥ 200) spectral	high SFS	high LBFS
sparse ($\leq$ 30 %)	**100**	2,98	1,78	10,57	3,68	1,97	58,85	4,24	2,20	-
	200	8,48	8,22	36,99	8,38	8,08	211,08	9,62	8,93	-
	300	16,69	17,58	83,08	18,00	16,55	513,76	18,18	16,58	-
	400	27,68	29,91	153,23	30,06	31,92	953,13	30,30	32,10	-
	500	38,78	44,35	209,87	47,77	47,33	1382,98	45,60	41,20	-
	600	50,28	53,66	277,90	59,06	55,47	1771,93	54,10	57,10	-
	700	67,02	73,45	383,13	72,54	75,64	2437,52	76,55	78,96	-
	800	98,54	98,29	526,48	94,76	98,96	3236,95	104,52	102,09	-
	900	114,36	124,67	616,90	121,75	122,12	4103,76	136,70	130,02	-
	1000	152,63	161,15	904,72	153,52	148,28	5047,28	189,63	184,12	-
normal ($>$ 30 % and $\leq$ 70%)	**100**	3,16	4,65	26,25	3,46	5,20	196,26	3,41	5,04	-
	200	11,04	18,58	108,28	12,96	19,92	942,65	14,43	20,08	-
	300	25,62	40,91	252,98	29,46	44,37	2098,60	30,71	45,09	-
	400	49,50	76,23	459,03	55,82	74,65	3833,16	56,85	79,34	-
	500	73,35	108,69	645,23	84,66	113,71	5659,31	84,77	110,84	-
	600	108,05	139,40	893,37	126,33	153,15	7437,49	126,89	148,99	-
	700	143,32	186,48	1247,81	164,40	196,33	10402,90	172,27	195,22	-
	800	193,45	253,49	1646,54	232,95	246,19	13920,20	253,77	255,05	-
	900	254,46	307,13	2131,64	317,26	309,65	17909,20	310,84	326,79	-
	1000	331,47	408,70	2856,86	383,54	376,66	22601,10	442,26	499,45	-
dense ($>$ 70 %)	**100**	3,87	6,81	66,58	3,89	7,72	493,64	3,89	7,78	-
	200	16,37	27,38	285,67	16,08	30,01	2126,32	16,95	31,57	-
	300	38,64	61,59	633,54	40,14	65,96	4904,51	38,32	69,41	-
	400	77,00	112,23	1165,52	76,81	114,90	9114,09	77,66	121,97	-
	500	122,27	158,87	1691,87	122,57	163,62	13693,00	114,96	161,89	-
	600	174,42	211,88	2349,12	173,31	210,19	18455,80	171,59	225,39	-
	700	273,01	291,58	3364,06	248,08	286,44	25932,80	245,26	299,84	-
	800	359,28	379,78	4493,35	339,09	373,69	34891,70	344,47	397,55	-
	900	489,78	487,85	5854,02	450,70	466,22	45060,20	450,22	519,41	-
	1000	663,46	642,58	8046,78	588,68	579,59	58410,50	707,10	775,99	-

Figure 1: (Average) Time performance of the algorithms (in milliseconds)

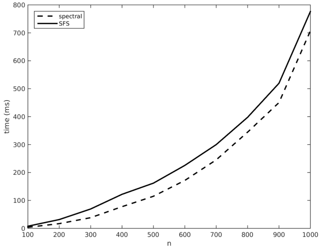
Performance chart ($n \leq 1000$)







(c) normal - low



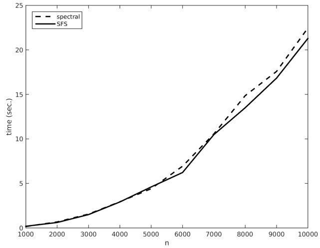
(d) dense - high

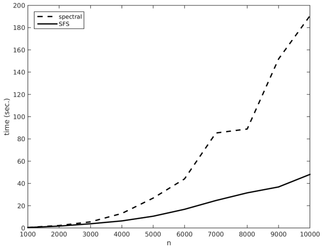
Performance table (large instances)

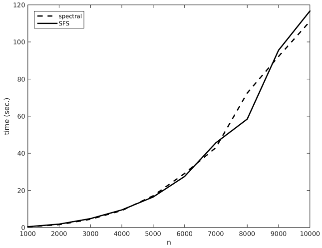
	# distinct values	**low** (≤ 50)			**medium** (> 50 and ≤ 200)			**high** (≥ 200)		
# nonzero entries	**algorithms** / **n**	**spectral**	**SFS**	**LBFS**	**spectral**	**SFS**	**LBFS**	**spectral**	**SFS**	**LBFS**
sparse (≤ 30 %)	**1000**	0,16	0,19	-	0,16	0,16	-	0,17	0,18	-
	2000	0,68	0,62	-	0,72	0,7	-	0,76	0,62	-
	3000	1,56	1,5	-	1,95	1,58	-	1,95	1,48	-
	4000	2,94	2,92	-	3,6	2,57	-	3,58	2,81	-
	5000	4,41	4,61	-	5,56	4,03	-	6,09	4,38	-
	6000	6,94	6,23	-	9,93	6,52	-	10,87	6,72	-
	7000	10,56	10,48	-	20,98	10,32	-	20,73	8,75	-
	8000	14,86	13,5	-	18,24	10,67	-	21,03	11,63	-
	9000	17,58	16,83	-	26,38	13,75	-	31,66	13,97	-
	10000	22,46	21,28	-	45,32	18,11	-	32,87	16,18	-
normal (> 30 % and ≤ 70%)	**1000**	0,32	0,4	-	0,45	0,41	-	0,45	0,46	-
	2000	1,53	1,8	-	2,2	1,67	-	1,99	1,71	-
	3000	4,42	4,77	-	5,49	3,77	-	5,74	3,64	-
	4000	9,13	9,46	-	13,04	6,33	-	14,22	6,54	-
	5000	17,08	16,45	-	26,85	10,55	-	26,33	10,77	-
	6000	29,09	27,48	-	44,08	16,76	-	43,07	18,11	-
	7000	43,05	45,63	-	85,31	24,65	-	68,86	21,71	-
	8000	72,48	58,42	-	88,91	31,54	-	86,72	30,49	-
	9000	92,18	95,53	-	151,81	36,85	-	116,02	36,87	-
	10000	111,08	116,67	-	190,55	48,09	-	155,1	43,41	-
dense (> 70 %)	**1000**	0,62	0,67	-	0,62	0,6	-	0,6	0,63	-
	2000	3,3	2,95	-	3,59	2,26	-	3,62	2,38	-
	3000	10,46	8,43	-	11,65	4,99	-	11,61	5,51	-
	4000	25,64	16,75	-	27,53	9,38	-	26,62	9,92	-
	5000	43,85	29,4	-	51,63	15,22	-	51,03	15,89	-
	6000	104,47	59,28	-	101,14	22,69	-	92,41	26,09	-
	7000	121,14	91,75	-	166,53	38,52	-	142,65	31,19	-
	8000	220,08	129,7	-	219,71	40,28	-	216,43	43,31	-
	9000	284,63	175,07	-	331,37	52,81	-	293,18	52,44	-
	10000	383,98	248,97	-	423,32	65,31	-	411,29	64,93	-

Figure 2: (Average) Time performance of the algorithms (in seconds)

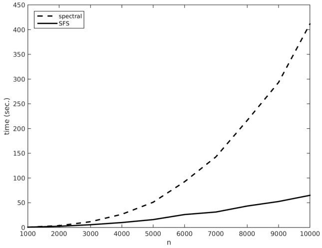
Performance chart (large instances)







(c) normal - low



(d) dense - high

Conclusions

- **Lex-BFS** is used to recognize unit interval graphs (3 sweeps, Corneil'04), cographs (2 sweeps, Bretscher & al.'08), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...

Conclusions

- **Lex-BFS** is used to recognize unit interval graphs (3 sweeps, Corneil'04), cographs (2 sweeps, Bretscher & al.'08), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- New **weighted graph search** algorithm: **SFS (Similarity-First Search)**.

  Very simple algorithm: conceptually and to implement:

  CRAN Package SFS available at the *R* platform.

  SFS permits to recognize Robinsonian matrices. Other applications?

Conclusions

- **Lex-BFS** is used to recognize unit interval graphs (3 sweeps, Corneil'04), cographs (2 sweeps, Bretscher & al.'08), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- New **weighted graph search** algorithm: **SFS (Similarity-First Search)**.

  Very simple algorithm: conceptually and to implement:

  CRAN Package SFS available at the *R* platform.

  SFS permits to recognize Robinsonian matrices. Other applications?
- **Robinsonian matrices** are matrix analogues of **unit interval graphs**.

  Investigate other matrix analogues, e.g., for interval graphs.

Conclusions

- **Lex-BFS** is used to recognize unit interval graphs (3 sweeps, Corneil'04), cographs (2 sweeps, Bretscher & al.'08), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- New **weighted graph search** algorithm: **SFS (Similarity-First Search)**.

  Very simple algorithm: conceptually and to implement:

  CRAN Package SFS available at the *R* platform.

  SFS permits to recognize Robinsonian matrices. Other applications?
- **Robinsonian matrices** are matrix analogues of **unit interval graphs**.

  Investigate other matrix analogues, e.g., for interval graphs.
- **'Chordal' matrices:** defined by existence of a **perfect elimination ordering** π: $A_{yz} \geq \min\{A_{xy}, A_{xz}\}$ if $x <_\pi y <_\pi z$

  Characterization by excluded 'weighted chordless cycles'.

[L-Tanigawa'17]

Based on papers

M. Laurent and M. Seminaroti.
The quadratic assignment problem is easy for Robinsonian matrices with Toeplitz structure.
Operations Research Letters, 2015.

M. Laurent and M. Seminaroti.
A Lex-BFS-based recognition algorithm for Robinsonian matrices.
Proceedings of CIAC 2015 & Discrete Applied Mathematics, 2017.

M. Laurent and M. Seminaroti.
Similarity-First Search: a new algorithm with application to Robinsonian matrix recognition.
SIAM Journal on Discrete Mathematics, 2017.

M. Seminaroti.
Combinatorial Algorithms for the Seriation Problem.
PhD thesis, Tilburg University, December 2016.