Ordering Robinsonian matrices with graph algorithms

Monique Laurent

Graph Theory in Paris – 23 November 2018

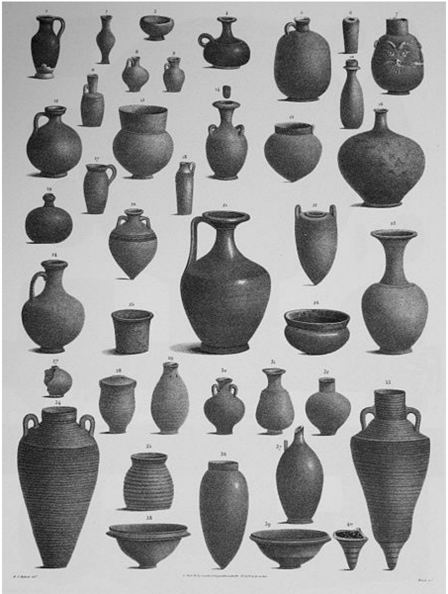
Based on joint works with Matteo Seminaroti

Plan of the talk

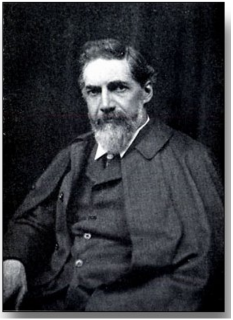
- Ordering similarity matrices: the seriation problem
- Numerical algorithm: the spectral approach
- Combinatorial algorithms: links to (unit interval) graphs
- Graph search: Lexicographic Breadth-First Search (Lex-BFS) (and unit interval graphs)
- New weighted graph search: Similarity-First Search (SFS) (and Robinson matrices)
- Combinatorial obstructions

The seriation problem

Motivation: Archeology

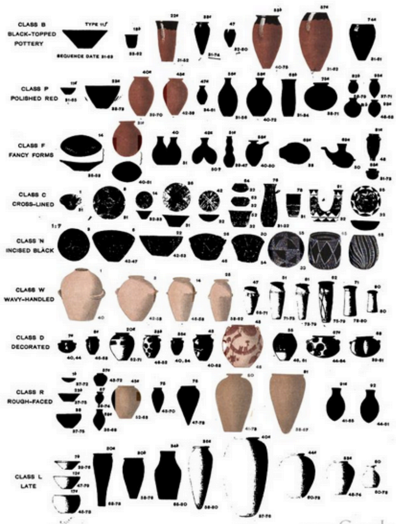


Sequence dating



Sir William Matthew Flinders Petrie (1853-1942)

1:14 COMMONEST TYPES OF PREHISTORIC POTTERY.



DIOSPOLIS PARVA

THE CEMETERIES OF ABADIYEH AND HU

1898-9

BY

W. M. FLINDERS PETRIE

HON. D.C.L., LITT.D., LL.D., PH.D.,

EDWARDS PROFESSOR OF EGYPTOLOGY, UNIVERSITY COLLEGE, LONDON;
MEMBER OF THE IMPERIAL GERMAN ARCHAEOLOGICAL INSTITUTE;
CORRESPONDING MEMBER OF SOCIETY OF ANTHROPOLOGY, BERLIN;
MEMBER OF THE SOCIETY OF NORTHERN ANTIQUARIES.

With Chapters by

A. C. MACE

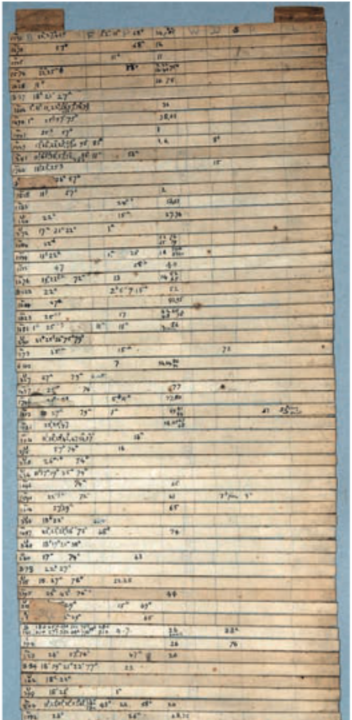
SPECIAL EXTRA PUBLICATION OF

THE EGYPT EXPLORATION FUND

PUBLISHED BY ORDER OF THE COMMITTEE

LONDON
SOLD AT
THE OFFICES OF THE EGYPT EXPLORATION FUND, 37, GREAT RUSSELL STREET, W.C.
AND AT 59, TEMPLE STREET, BOSTON, MASS., U.S.A.
AND BY KEGAN PAUL, TRENCH, TRÜBNER & CO., PATERNOSTER HOUSE, CHARING CROSS ROAD, W.C.
B. QUARITCH, 15, PICCADILLY, W.; ASHER & Co., 13, BEDFORD STREET, COVENT GARDEN, W.C.

1901



Paper-slips of Petrie

Seriation and the Consecutive Ones Property (C1P)

Try to order the graves so that 'similar' graves are close to each other in the ordering.

Seriation and the Consecutive Ones Property (C1P)

Try to order the graves so that 'similar' graves are close to each other in the ordering.

$$\begin{array}{c} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{pmatrix} & 1 & & \\ 1 & & 1 & 1 \\ & & 1 & 1 \\ & & 1 & \\ 1 & 1 & 1 & 1 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{pmatrix} & 1 & & \\ 1 & 1 & 1 & 1 \\ 1 & & 1 & 1 \\ & & 1 & 1 \\ & & 1 & \end{pmatrix} \end{array}$$

Matrix with C1P
P

Petrie matrix
ΠP

Permute the rows of P so that the ones are consecutive in its columns.

The approach of **Petrie** is based on the *presence/absence* of pottery types in the graves.

W.S. Robinson (1951) also uses the *frequency* of pottery types in the graves.

AMERICAN ANTIQUITY

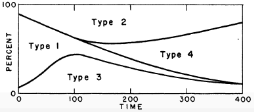
VOL. XVI APRIL, 1951 No. 4

A METHOD FOR CHRONOLOGICALLY ORDERING ARCHAEOLOGICAL DEPOSITS*

W. S. ROBINSON

THEORY

THE statistical technique of this paper is based upon the empirically established fact that over the course of time pottery types come into and go out of general use by a given group of people. It is further based upon the established fact that in cultures where chronology has been determined the differential use of types takes on a form illustrated in Figure 89. The data of this diagram are hypothetical, the purpose being merely to illustrate the present discussion.



into use at the beginning of the period, attains its greatest popularity around the year 100, and thereafter declines in importance. Type 4, on the other hand, first makes its appearance around the year 100, and increases in importance throughout the remaining years shown on the diagram.

The fact that types come into and go out of use in the lenticular fashion shown in Figure 89 has important implications for the archaeologist. Suppose he has a number of deposits, and that these deposits represent different points of time in the development of a people. Assuming that he already has the information given in Figure 89, what can he tell about the properties of these deposits? Reference to the figure will show that a deposit representing an early stage in this culture will have in it a preponderance of pottery of type 1, with small percentages of types 2 and 3. A deposit which represents an intermediate stage, on the other hand, will show a largest percentage of pottery of type 2, a somewhat

The **dissimilarity** measure $d(G_i, G_j)$ between two graves G_i, G_j is the ℓ_1-distance between their pottery-types frequency vectors.

$\rightsquigarrow$ their **similarity** measure (*agreement coefficient*) is $\mathrm{C} - d(G_i, G_j)$.

TABLE 17. PERCENTAGES OF EIGHT TYPES OF POTTERY IN THREE STRATIFIED TRENCHES

	Deposits							
Type	IIA	IIB	IIC	IA	IB	IIIA	IIIB	IIIC
1	24.0	1.4	.2	11.3	.3	29.6	54.3	.0
2	66.8	.9	.0	.0	.0	.0	3.5	.0
3	1.3	.0	.2	3.8	.2	14.1	14.0	6.6
4	.0	.0	.0	1.3	.2	.0	1.8	3.3
5	.0	.0	.0	3.3	.5	.0	5.3	5.5
6	4.0	.0	.0	24.9	1.4	7.0	7.0	27.5
7	.0	97.7	99.3	52.6	97.4	.0	12.3	57.1
8	3.9	.0	.3	2.8	.0	49.3	1.8	.0
	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0

TABLE 20. AGREEMENT COEFFICIENTS FOR THREE STRATIFIED TRENCHES—3RD ORDER

	IIA	IIIA	IIIB	IA	IIIC	IB	IIB	IIC
IIA		66 −	69 +	39 +	11 +	4 −	5 +	1
IIIA	66		101 +	50 +	27 +	4 +	3 +	1
IIIB	69 +	101		82 +	66 +	30 +	29 +	26
IA	39 +	50 +	82		172 +	110 +	108 +	107
IIIC	11 +	27 +	66 +	172		119 +	114 −	115
IB	4 o	4 +	30 +	110 +	119		195 −	196
IIB	5 −	3 +	29 +	108 +	114 +	195		196
IIC	1 o	1 +	26 +	107 +	115 +	196 o	196	
	195	252	403	668	624	658	650	642

W.S. Robinson (1951):

Order the graves, given by their pairwise similarities, in such a way that similar graves are placed close to each other in the ordering.

Seriation and Robinson similarity matrices

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.*

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c} \\ G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_5 & G_2 & G_3 & G_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Robinson matrix

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c c} & \begin{array}{ccccc} G_1 & G_2 & G_3 & G_4 & G_5 \end{array} \\ \begin{array}{c} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{array} & \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 3 & 2 & 1 & 3 \\ 0 & 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 3 & 1 & 1 & 4 \end{pmatrix} \end{array} \qquad \begin{array}{c c} & \begin{array}{ccccc} G_1 & G_5 & G_2 & G_3 & G_4 \end{array} \\ \begin{array}{c} G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Robinsonian matrix
A

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

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_2 & G_3 & G_4 & G_5 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 3 & 2 & 1 & 3 \\ 0 & 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 3 & 1 & 1 & 4 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_5 & G_2 & G_3 & G_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

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

Theorem (Kendall 1969)

- *For P $0/1$-**valued**:* P *is* **Petrie** $\iff$ PP^T *is* **Robinson**.

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c} \\ \begin{matrix} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{matrix} \end{array} \begin{array}{c} \begin{matrix} G_1 & G_2 & G_3 & G_4 & G_5 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 3 & 2 & 1 & 3 \\ 0 & 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 3 & 1 & 1 & 4 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ \begin{matrix} G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{matrix} \end{array} \begin{array}{c} \begin{matrix} G_1 & G_5 & G_2 & G_3 & G_4 \end{matrix} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Robinsonian matrix
A

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

Theorem (Kendall 1969)

- *For* P $0/1$-**valued***:* ΠP *is* **Petrie** $\iff$ $\Pi P P^T \Pi^T$ *is* **Robinson**.

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c|c} & \begin{array}{ccccc} G_1 & G_2 & G_3 & G_4 & G_5 \end{array} \\ \begin{array}{c} G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{array} & \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 3 & 2 & 1 & 3 \\ 0 & 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 3 & 1 & 1 & 4 \end{pmatrix} \end{array} \qquad \begin{array}{c|c} & \begin{array}{ccccc} G_1 & G_5 & G_2 & G_3 & G_4 \end{array} \\ \begin{array}{c} G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} & \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

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

Theorem (Kendall 1969)

- *For* P $0/1$-**valued***:* ΠP *is* **Petrie** $\iff$ $\Pi P P^T \Pi^T$ *is* **Robinson***.*
- P *has* **unimodal columns** $\iff P \circ P^{\mathsf{T}} := (\sum_z \min\{P_{xz}, P_{yz}\})_{x,y}$ *is* **Robinson***.*

Seriation and Robinson similarity matrices

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.*

$$\begin{array}{c} \\ G_1 \\ G_2 \\ G_3 \\ G_4 \\ G_5 \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_2 & G_3 & G_4 & G_5 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 3 & 2 & 1 & 3 \\ 0 & 2 & 2 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 1 & 3 & 1 & 1 & 4 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ G_1 \\ G_5 \\ G_2 \\ G_3 \\ G_4 \end{array} \begin{array}{c} \begin{array}{ccccc} G_1 & G_5 & G_2 & G_3 & G_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 4 & 3 & 2 & 1 \\ 0 & 3 & 3 & 2 & 1 \\ 0 & 2 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Robinsonian matrix A

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

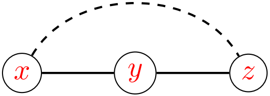
Theorem (Kendall 1969)

- *For P $0/1$-**valued**:* ΠP *is* **Petrie** $\Longleftrightarrow$ $\Pi P P^T \Pi^T$ *is* **Robinson**.

  ΠP *has* **unimodal columns** $\Longleftrightarrow$ $\Pi P \circ P^T \Pi^T$ *is* **Robinson**.

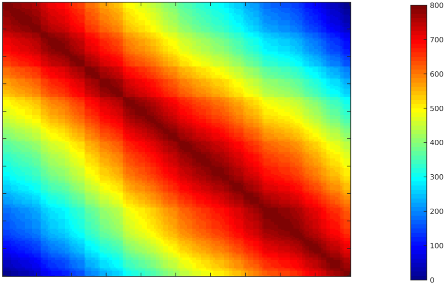
Robinson(ian) similarity matrix

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



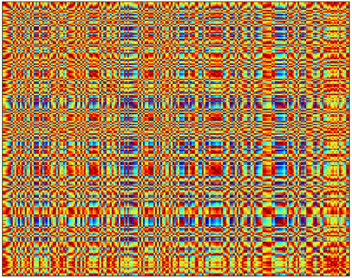
$$A_{xz} \leq \min\{A_{xy}, A_{yz}\}$$

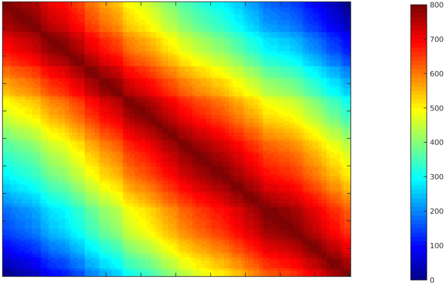
$$\forall\, 1 \leq x < y < z \leq n$$



Robinson(ian) similarity matrix

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

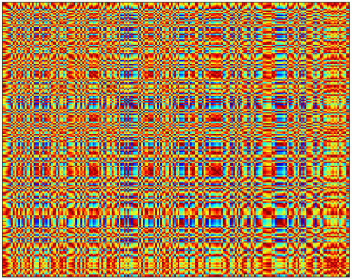


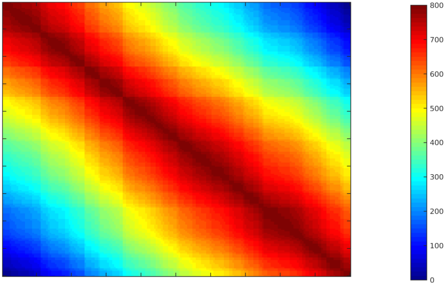


$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

Robinson(ian) similarity matrix

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



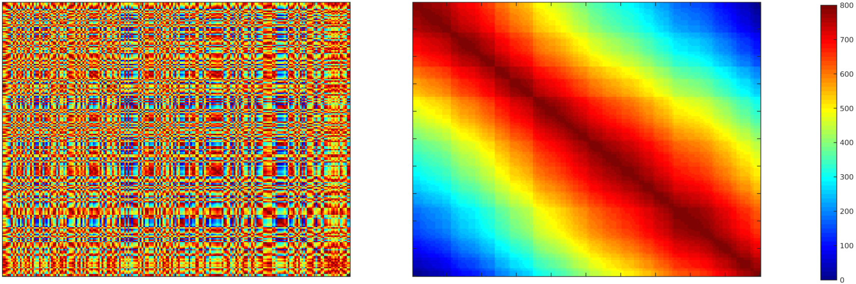


$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

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

Robinson(ian) similarity matrix

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



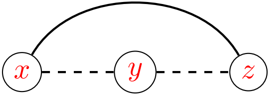
$A \in \mathcal{S}^n$ is a **Robinsonian similarity** if there exists a permutation π such that $\Pi A \Pi^T = A^\pi := \left(A_{\pi(x),\pi(y)}\right)_{x,y}$ is a **Robinson similarity**.

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

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

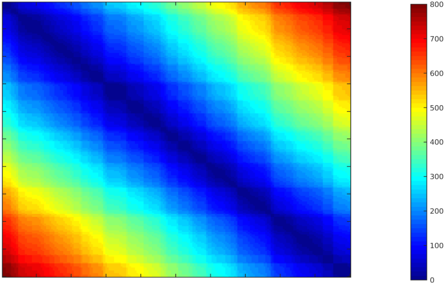
Robinson(ian) dissimilarity matrix

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



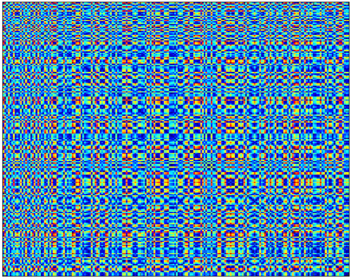
$$D_{xz} \geq \max\{D_{xy}, D_{yz}\}$$

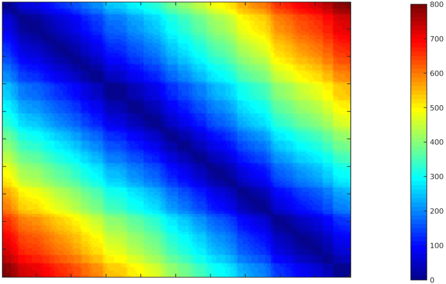
$$\forall\ 1 \leq x < y < z \leq n$$



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

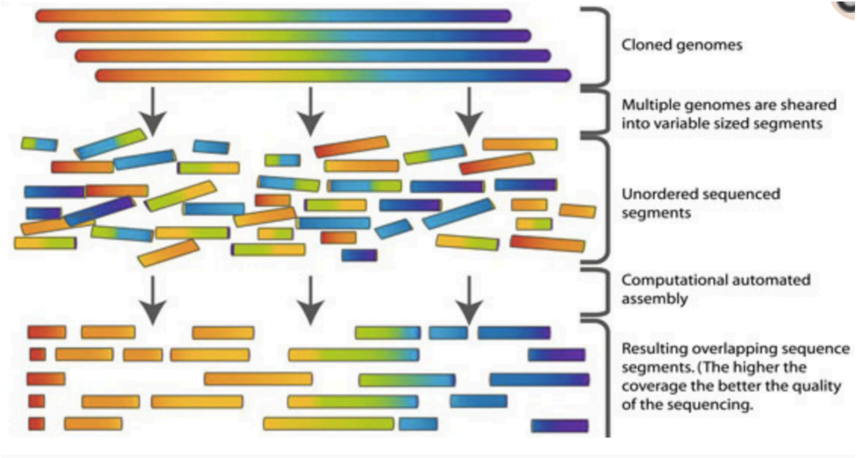
*Given $A \in \mathcal{S}^n$, find a permutation π (**Robinson ordering**) for which A^π is Robinson, or decide that none exists.*

There are efficient algorithms:

- Numerical algorithm: spectral method
- Combinatorial algorithms: via interval graphs and graph search

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

DNA sequencing



Seriation, quadratic assignment and the spectral algorithm

Seriation and Quadratic Assignment

A: similarity matrix $\qquad$ D: dissimilarity matrix

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

Seriation and Quadratic Assignment

A: similarity matrix $\qquad$ D: dissimilarity matrix

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

- $D = (|x - y|)$ $\rightsquigarrow$ 1-SUM problem
- $D = ((x - y)^2)$ $\rightsquigarrow$ 2-SUM problem

NP-hard problems for general A [George-Pothen'97]

Seriation and Quadratic Assignment

A: similarity matrix $\qquad$ D: dissimilarity matrix

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

- $D = (|x - y|)$ $\rightsquigarrow$ 1-SUM problem
- $D = ((x - y)^2)$ $\rightsquigarrow$ 2-SUM problem

  NP-hard problems for general A [George-Pothen'97]

- Note: in both cases D is a **Robinson dissimilarity** and D is **Toeplitz**: constant entries on each diagonal.

Seriation and Quadratic Assignment

A: similarity matrix $\qquad$ D: dissimilarity matrix

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

- $D = (|x - y|)$ $\rightsquigarrow$ 1-SUM problem
- $D = ((x - y)^2)$ $\rightsquigarrow$ 2-SUM problem

  NP-hard problems for general A [George-Pothen'97]

- Note: in both cases D is a **Robinson dissimilarity** and D is **Toeplitz**: constant entries on each diagonal.

Theorem (L-Seminaroti'15)

If D is a **Toeplitz Robinson dissimilarity** *and A is a* **Robinsonian similarity** *then any Robinson ordering π of A is an optimal solution.*

Seriation and Quadratic Assignment

A: similarity matrix $\qquad$ D: dissimilarity matrix

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

- $D = (|x-y|)$ $\rightsquigarrow$ 1-SUM problem
- $D = ((x-y)^2)$ $\rightsquigarrow$ 2-SUM problem

  NP-hard problems for general A [George-Pothen'97]

- Note: in both cases D is a **Robinson dissimilarity** and D is **Toeplitz**: constant entries on each diagonal.

Theorem (L-Seminaroti'15)

If D is a **Toeplitz Robinson dissimilarity** *and A is a* **Robinsonian similarity** *then any Robinson ordering π of A is an optimal solution. Hence QAP(A, D) is polynomial time solvable.*

Extending a result of [Fogel, Jenatton, Bach, Aspremont 2014]

Idea behind this result

For any permutation π:

$$\sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)} \geq \sum_{x,y=1}^{n} A_{xy} D_{xy}$$

when:

$$A = \begin{pmatrix} * & & \leftarrow & \\ & * & & \downarrow \\ \uparrow & & * & \\ & \rightarrow & & * \end{pmatrix}, \qquad D = \begin{pmatrix} * & & \rightarrow & \\ & * & & \uparrow \\ \downarrow & & * & \\ & \leftarrow & & * \end{pmatrix} \ \text{Toeplitz}$$

Idea behind this result

For any permutation π:

$$\sum_{x,y=1}^{n} A_{xy} D_{\pi(x)\pi(y)} \geq \sum_{x,y=1}^{n} A_{xy} D_{xy}$$

when:

$$A = \begin{pmatrix} * & & \leftarrow & \\ & * & & \downarrow \\ \uparrow & & * & \\ & \rightarrow & & * \end{pmatrix}, \qquad D = \begin{pmatrix} * & & \rightarrow & \\ & * & & \uparrow \\ \downarrow & & * & \\ & \leftarrow & & * \end{pmatrix} \ \text{Toeplitz}$$

This is the analogous for matrices of the **rearrangement inequality**:

$$\sum_{x=1}^{n} a_x d_{\pi(x)} \geq \sum_{x=1}^{n} a_x d_x$$

when:

$$a_1 \geq \cdots \geq a_n$$
$$d_1 \leq \cdots \leq d_n$$

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow \quad$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow \quad$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow \quad$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

Idea: "Relax" 2-SUM: $\boxed{\min_\pi \sum_{x,y=1}^n A_{xy}(\pi(x) - \pi(y))^2}$ by

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

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow \quad$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

Idea: "Relax" 2-SUM: $\boxed{\min_\pi \sum_{x,y=1}^n A_{xy}(\pi(x) - \pi(y))^2}$ by

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

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then L_A has a* **monotone** *Fiedler vector.*

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

Idea: "Relax" 2-SUM: $\boxed{\min_\pi \sum_{x,y=1}^n A_{xy}(\pi(x) - \pi(y))^2}$ by

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

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then L_A has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$.*
   If A is Robinson(ian) then $\lambda_2(L_A) > \mathbf{0}$ and $\lambda_2(L_A)$ is **simple**.

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0$ $\rightsquigarrow$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

Idea: "Relax" 2-SUM: $\boxed{\min_\pi \sum_{x,y=1}^n A_{xy}(\pi(x) - \pi(y))^2}$ by

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

Theorem (Atkins-Boman-Hendrickson 1998)

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

The spectral algorithm to recognize Robinsonian matrices

Similarity matrix $A \geq 0 \quad \rightsquigarrow \quad$ **Laplacian matrix:** $L_A = \mathrm{Diag}(Ae) - A$.

- $\lambda_1(L_A) = 0$, with eigenvector the all-ones vector e.
- $\lambda_2(L_A)$ is the **Fiedler value**, its eigenvectors are the **Fiedler vectors**.

Idea: "Relax" 2-SUM: $\boxed{\min_\pi \sum_{x,y=1}^n A_{xy}(\pi(x) - \pi(y))^2}$ by

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

Theorem (Atkins-Boman-Hendrickson 1998)

1. *If A is Robinson then L_A has a* **monotone** *Fiedler vector.*
2. *Assume A is irreducible with $\min_{i,j} A_{ij} = 0$.*
   If A is Robinson(ian) then $\lambda_2(L_A) > \mathbf{0}$ and $\lambda_2(L_A)$ is **simple**.
3. *If the Fiedler vector v has* **no repeated entries**, *then a permutation π orders v monotonically $\iff \pi$ is a Robinson ordering of A.*
   Else **recurse** *on the submatrices indexed by the repeated entries.*

Combinatorial algorithms via

(unit) interval graphs

Robinsonian matrices, interval graphs and C1P

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 (Fulkerson-Gross'65, Mirkin-Rodin'84)

The following are equivalent:

1. *A is a Robinsonian similarity*
2. *the intersection graph of $\mathcal{B}$ is an* **interval graph**

Robinsonian matrices, interval graphs and C1P

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 (Fulkerson-Gross'65, Mirkin-Rodin'84)

The following are equivalent:

1. A *is a Robinsonian similarity*
2. *the intersection graph of* $\mathcal{B}$ *is an* **interval graph**

   $\Longleftrightarrow$ *its max.cliques/vertices incidence matrix has* **C1P***.*

Robinsonian matrices, interval graphs and C1P

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 (Fulkerson-Gross'65, Mirkin-Rodin'84)

The following are equivalent:

1. A *is a Robinsonian similarity*
2. *the intersection graph of* $\mathcal{B}$ *is an* **interval graph**

   $\iff$ *its max.cliques/vertices incidence matrix has* **C1P**.
3. *the vertices/balls incidence matrix has* **C1P**

   ($\rightsquigarrow$ *the ball hypergraph* $(V, \mathcal{B})$ *is an* **interval hypergraph***)*

Robinsonian matrices, interval graphs and C1P

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 (Fulkerson-Gross'65, Mirkin-Rodin'84)

The following are equivalent:

1. A *is a Robinsonian similarity*
2. *the intersection graph of* $\mathcal{B}$ *is an* **interval graph**

   $\Longleftrightarrow$ *its max.cliques/vertices incidence matrix has* **C1P**.
3. *the vertices/balls incidence matrix has* **C1P**

   ($\rightsquigarrow$ *the ball hypergraph* $(V, \mathcal{B})$ *is an* **interval hypergraph***)*

Theorem (Booth-Lueker 1976)

One can test whether a matrix $M \in \{0,1\}^{p\times q}$ *with* m *ones has* **C1P** *in* $O(p+q+m)$ *(using PQ-trees).*

Existing recognition algorithms for Robinsonian matrices

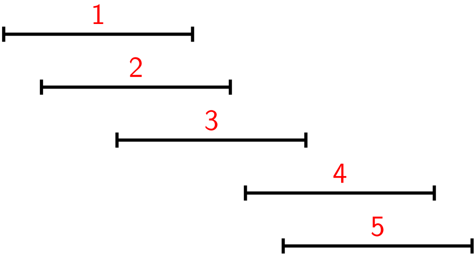
	Year	Complexity	Subroutine	Paradigm
Mirkin & Rodin	1984	$O(n^4)$	PQ-trees	interval hypergraphs
Chepoi & Fichet	1997	$O(n^3)$	PQ-trees	interval hypergraphs
Préa & Fortin	2014	$O(n^2)$		interval graphs
Atkins et al.	1998	$O(n(T(n) + n \log n))$	eigenvalues	Fiedler vector
Laurent & Seminaroti	2015	$O(L(m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

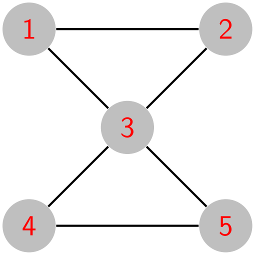
n: size of A; m : # of nonzero entries of A; L : # of distinct values of A.

Unit interval graphs and binary Robinsonian matrices

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

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





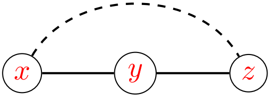
Unit interval graphs and binary Robinsonian matrices

Theorem (Looges-Olariu 1993)

G *is a* **unit interval graph** $\iff$ *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:



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

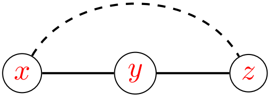
Unit interval graphs and binary Robinsonian matrices

Theorem (Looges-Olariu 1993)

G is a **unit interval graph** $\iff$ *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:



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

Fact (Roberts 1969)

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

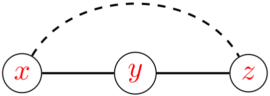
Unit interval graphs and binary Robinsonian matrices

Theorem (Looges-Olariu 1993)

G is a **unit interval graph** $\iff$ *there exists a linear order π of the vertices satisfying the* **3-point condition***:*

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

Recall the Robinson (similarity) property:



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

Fact (Roberts 1969)

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

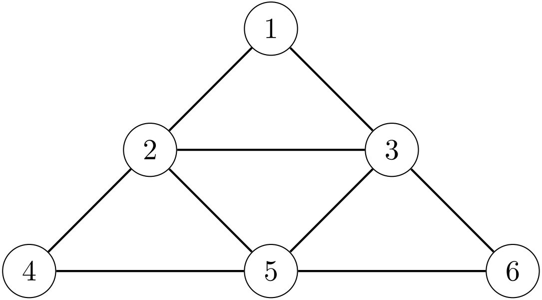
Theorem (Corneil 2004)

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)

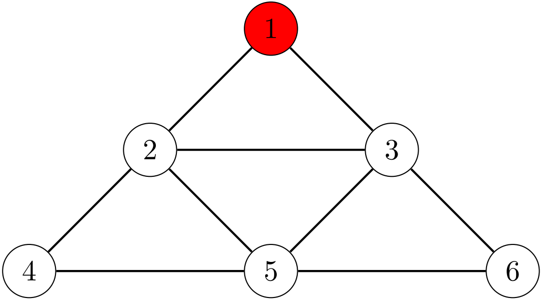
pivot

$Q:$

1	2	3	4	5	6

Graph search paradigm

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



visited vertices

unvisited vertices
(stored in a queue Q)

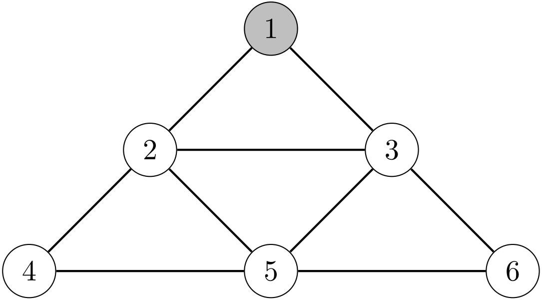
pivot

$Q:$

1	2	3	4	5	6

Graph search paradigm

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



visited vertices

unvisited vertices
(stored in a queue Q)

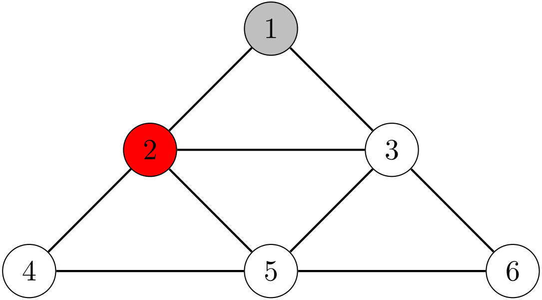
pivot

$Q:$

2	3	4	5	6

Graph search paradigm

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



visited vertices

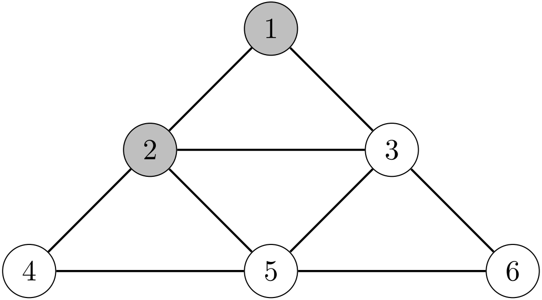
unvisited vertices
(stored in a queue Q)

pivot

Q :	2	3	4	5	6

Graph search paradigm

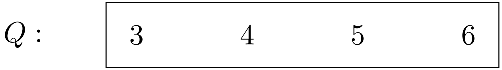
Given a graph $G = (V, E)$:



visited vertices

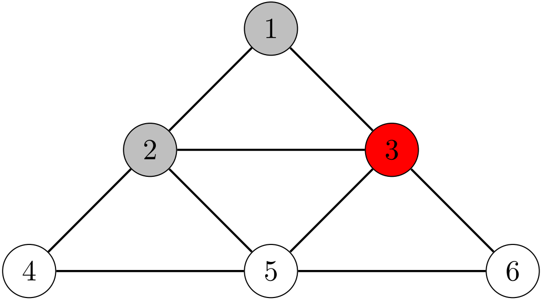
unvisited vertices
(stored in a queue Q)

pivot



Graph search paradigm

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



visited vertices

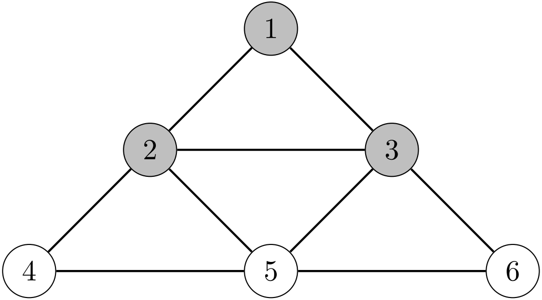
unvisited vertices
(stored in a queue Q)

pivot

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

Graph search paradigm

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



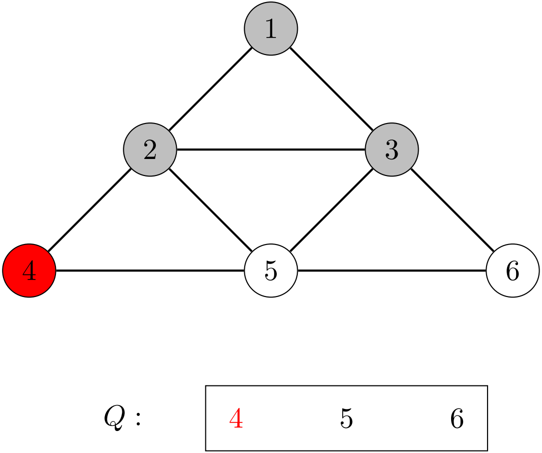
visited vertices

unvisited vertices
(stored in a queue Q)

pivot

Graph search paradigm

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



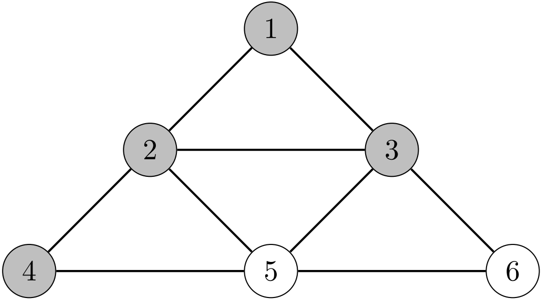
visited vertices

unvisited vertices
(stored in a queue Q)

pivot

Graph search paradigm

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



visited vertices

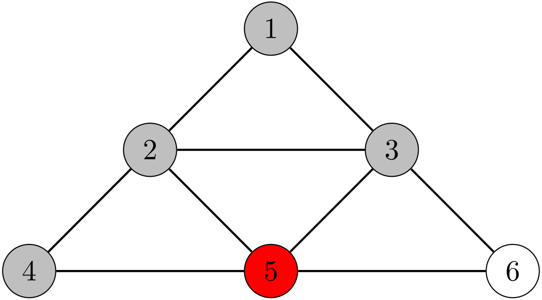
unvisited vertices
(stored in a queue Q)

pivot

Q :

Graph search paradigm

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



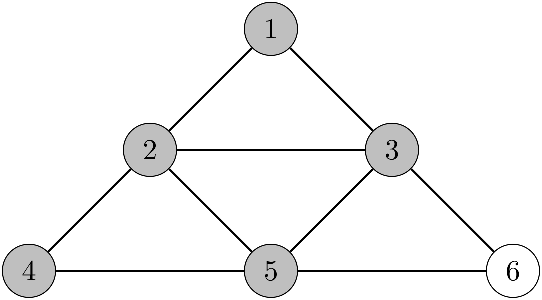
visited vertices

unvisited vertices
(stored in a queue Q)

pivot

Graph search paradigm

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



visited vertices

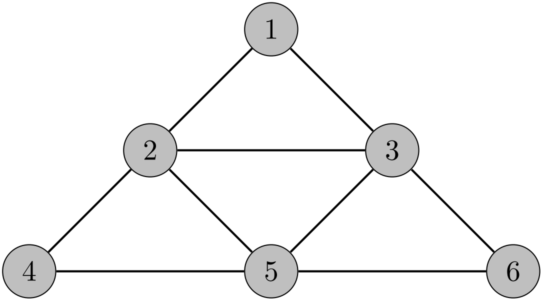
unvisited vertices
(stored in a queue Q)

pivot

Q : $\boxed{6}$

Graph search paradigm

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



visited vertices

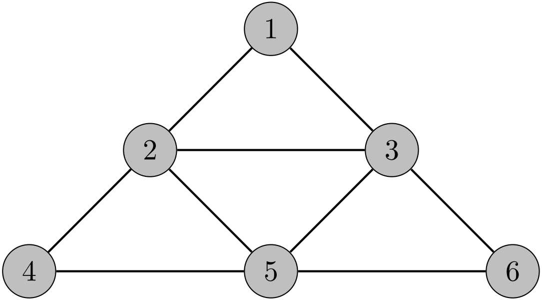
unvisited vertices
(stored in a queue Q)

pivot

Q : $\boxed{\varnothing}$

Graph search paradigm

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



visited vertices

unvisited vertices
(stored in a queue Q)

pivot

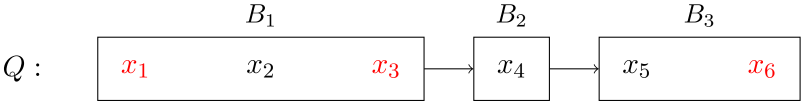
Different queue updates lead to different graph search algorithms:

- Breadth-First Search (BFS)
- Depth-First Search (DFS)
- Lexicographic Breadth-First Search (Lex-BFS)
  "Give the preference to vertices adjacent to vertices visited earlier."

Lex-BFS via partition refinement

Idea: Maintain (and refine) a **partition** of the queue Q.

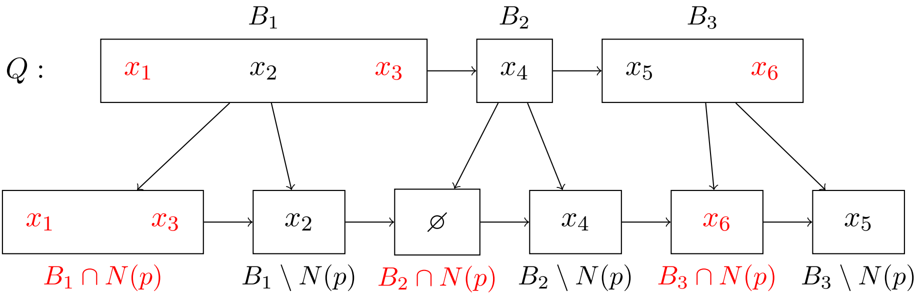
Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS via partition refinement

Idea: Maintain (and refine) a **partition** of the queue Q.

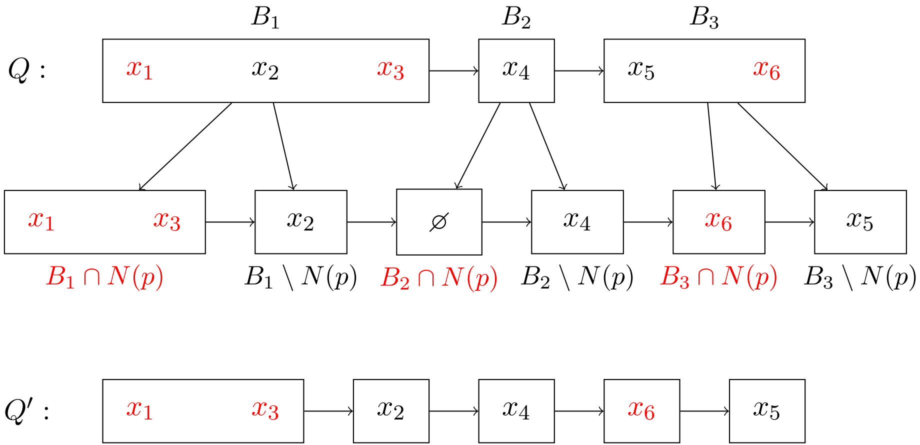
Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS via partition refinement

Idea: Maintain (and refine) a **partition** of the queue Q.

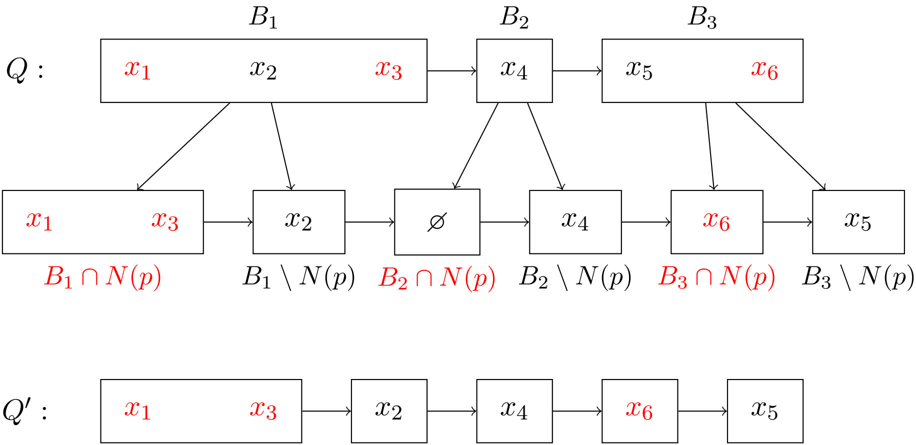
Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS via partition refinement

Idea: Maintain (and refine) a **partition** of the queue Q.

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

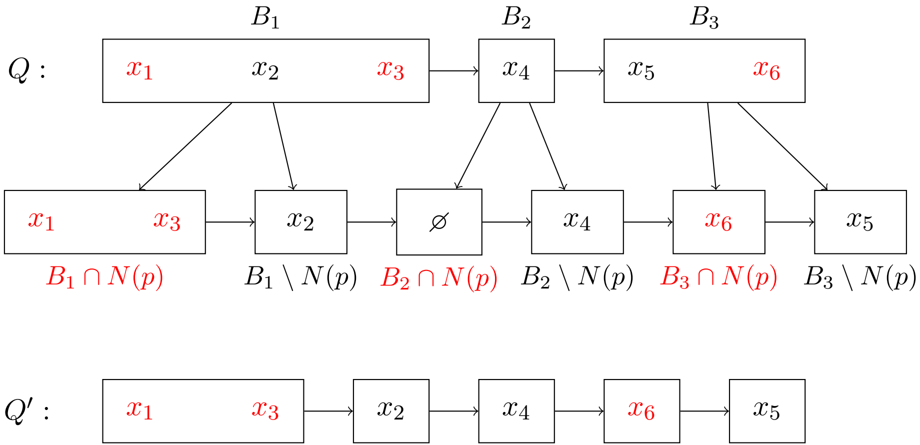


Lex-BFS runs in time $O(|V| + |E|)$ [Rose-Tarjan'75, Habib et al.'00]

Lex-BFS via partition refinement

Idea: Maintain (and refine) a **partition** of the queue Q.

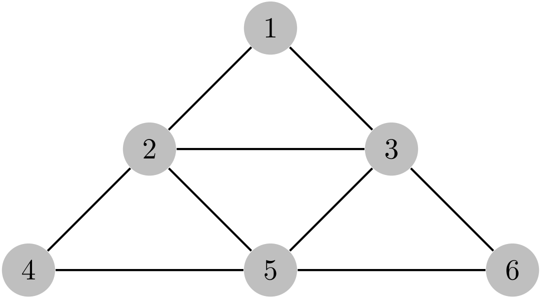
Let $N(p)$ denote the neighborhood of the current pivot p.



Lex-BFS$_+(G, \tau)$: Order vertices in the blocks using a **reference order** τ.

Example of Lex-BFS$_+$

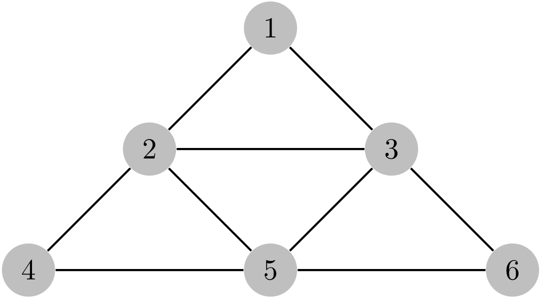
$\tau = (1, 2, 3, 4, 5, 6)$



1	2	3	4	5	6

Example of Lex-BFS$_+$

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

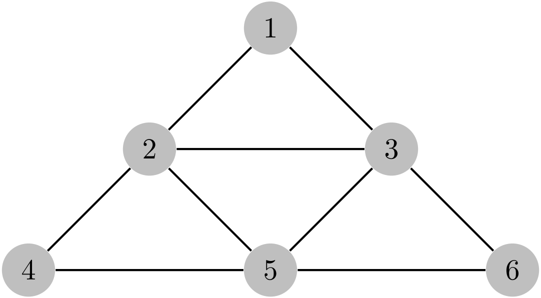


$$\boxed{1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6}$$

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

Example of Lex-BFS$_+$

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



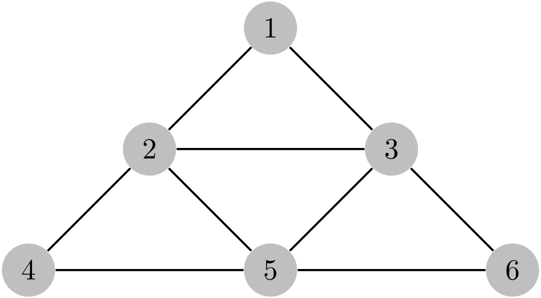
$\boxed{1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6}$

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

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

Example of Lex-BFS$_+$

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



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

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

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

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

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. $\sigma_+ =$ 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]

$\rightsquigarrow O(L(m+n))$

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]
$\rightsquigarrow O(L(m+n))$

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

Weighted graph search:

Similarity-First Search (SFS)

Similarity-First Search (SFS) for nonnegative A

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

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 $N(p) = \{x : A_{px} > 0\}$.

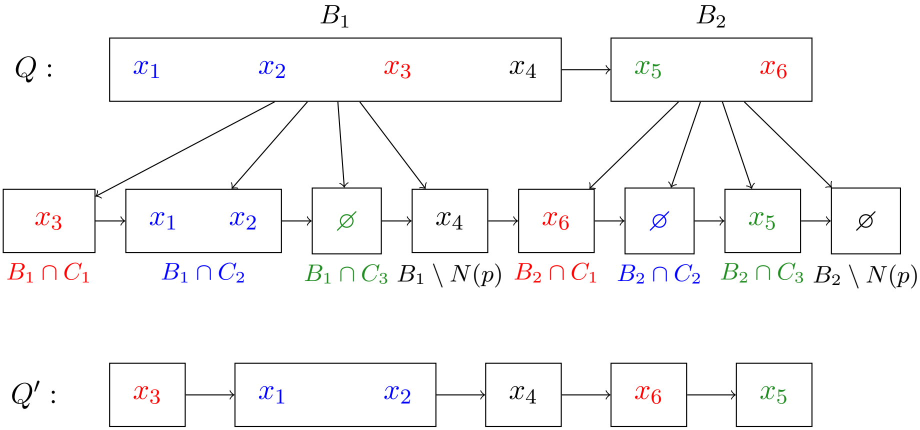
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \ldots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \ldots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \ldots$

$$Q: \quad \overset{B_1}{\boxed{x_1 \quad x_2 \quad x_3 \quad x_4}} \longrightarrow \overset{B_2}{\boxed{x_5 \quad x_6}}$$

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 $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \dots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \dots$

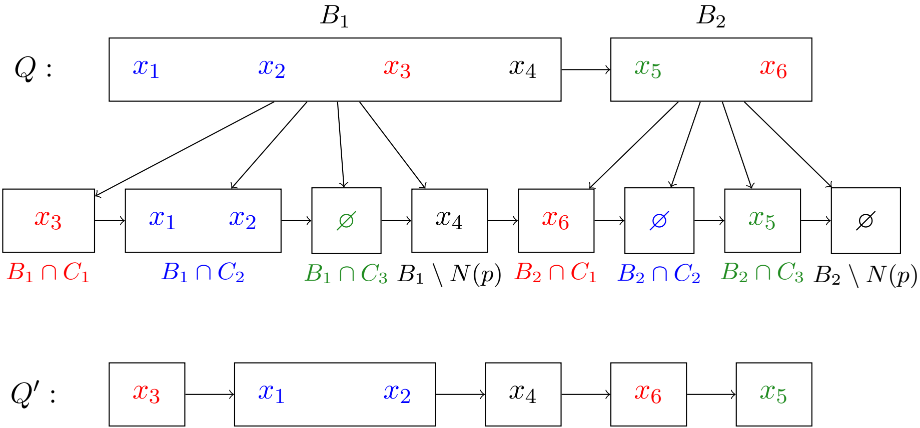


SFS runs in $O(n + m \log n)$ if A has m nonzero entries. [L-Seminaroti 17]

Similarity-First Search (SFS) for nonnegative A

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

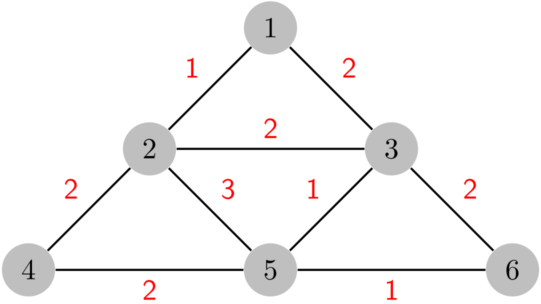
Consider the ordered **similarity partition** $(C_1, C_2, C_3, \dots)$ of $N(p)$, where $A_{px} = \alpha_1 > A_{py} = \alpha_2 > A_{pz} = \alpha_3 > \dots > 0 \;\; \forall x \in C_1, y \in C_2, z \in C_3, \dots$



SFS$_+(A, \tau)$**:** order the vertices in each block using a **reference order** τ

Example for SFS_+

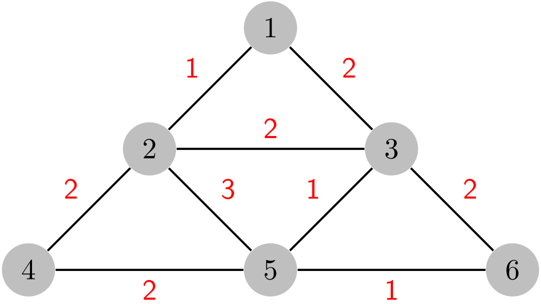
$\tau = (1, 2, 3, 4, 5, 6)$



1	2	3	4	5	6

Example for SFS_+

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

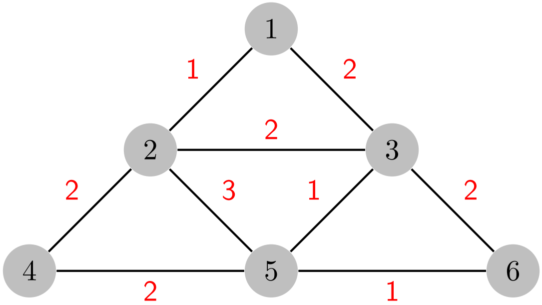


$$\boxed{1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6}$$

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

Example for SFS_+

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

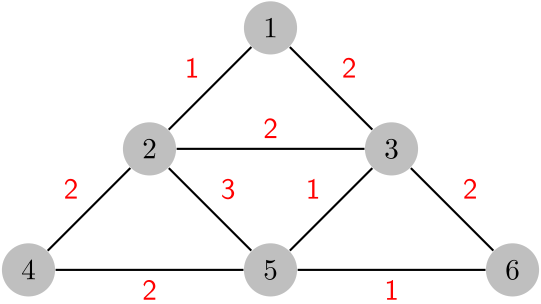


$\boxed{1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6}$ $\qquad$ $1 \quad 3 \quad \boxed{2} \ \boxed{6} \ \boxed{5} \ \boxed{4}$

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

Example for SFS$_+$

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



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

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

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** $i = 1, \ldots, n-2$
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**

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. $\quad \sigma_i = \mathsf{SFS}_+(A, \sigma_{i-1}^{-1})$
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.

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}.

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. *Then 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 σ = SFS(A) has **last vertex** *b.*

1. *Then 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.*

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$_+$

$$
\begin{array}{llllllll}
\boldsymbol{\sigma_0}: & u_1 & u_2 & u_3 & \cdots & u_{n-2} & u_{n-1} & a \\
\boldsymbol{\sigma_1}: & a & x_2 & x_3 & \cdots & x_{n-2} & x_{n-1} & b \\
\boldsymbol{\sigma_2}: & b & y_2 & y_3 & \cdots & y_{n-2} & y_{n-1} & a
\end{array}
$$

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$_+$

$$
\begin{array}{cccccccc}
\sigma_{\mathbf{0}}: & u_1 & u_2 & u_3 & \cdots & u_{n-2} & u_{n-1} & a \\
\sigma_{\mathbf{1}}: & a & x_2 & x_3 & \cdots & x_{n-2} & x_{n-1} & b \\
\sigma_{\mathbf{2}}: & b & y_2 & y_3 & \cdots & y_{n-2} & y_{n-1} & a \\
\sigma_{\mathbf{3}}: & a & a_1 & a_2 & \cdots & b_2 & b_1 & b
\end{array}
$$

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.

Anchor flipping property of SFS$_+$

$$
\begin{array}{cccccccc}
\sigma_{\mathbf{0}}: & u_1 & u_2 & u_3 & \cdots & u_{n-2} & u_{n-1} & \color{red}{a} \\
\sigma_{\mathbf{1}}: & \color{red}{a} & x_2 & x_3 & \cdots & x_{n-2} & x_{n-1} & \color{red}{b} \\
\sigma_{\mathbf{2}}: & \color{red}{b} & y_2 & y_3 & \cdots & y_{n-2} & y_{n-1} & \color{red}{a} \\
\sigma_{\mathbf{3}}: & \color{red}{a} & \color{blue}{a_1} & a_2 & \cdots & b_2 & \color{blue}{b_1} & \color{red}{b}
\end{array}
$$

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$_+$

$$
\begin{array}{cccccccc}
\sigma_{\mathbf{0}}: & u_1 & u_2 & u_3 & \cdots & u_{n-2} & u_{n-1} & a \\
\sigma_{\mathbf{1}}: & a & x_2 & x_3 & \cdots & x_{n-2} & x_{n-1} & b \\
\sigma_{\mathbf{2}}: & b & y_2 & y_3 & \cdots & y_{n-2} & y_{n-1} & a \\
\sigma_{\mathbf{3}}: & a & a_1 & a_2 & \cdots & b_2 & b_1 & b
\end{array}
$$

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.**

Obstructions for Robinsonian matrices

Certifying non-Robinsonian matrices

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.,

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

Certifying non-Robinsonian matrices

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.*

Certifying non-Robinsonian matrices

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.*

Definition

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

If such triple exists then A is not Robinsonian!

Certifying non-Robinsonian matrices

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.*

Definition

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

Theorem (L-Seminaroti-Tanigawa 2017)

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

Certifying non-Robinsonian matrices

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.*

Definition

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

Theorem (L-Seminaroti-Tanigawa 2017)

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

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

Certifying non-Robinsonian matrices

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.*

Definition

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

Theorem (L-Seminaroti-Tanigawa 2017)

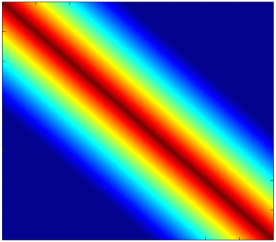
A is Robinsonian $\iff$ there does not exist a weighted asteroidal triple.

- Find a weighted asteroidal triple in $O(n^3)$: certifies A **not Robinsonian**.
- 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]

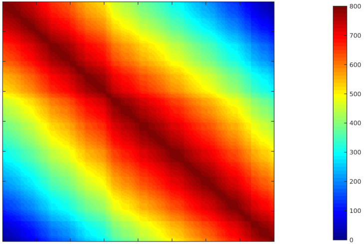
Computational experiments

Matteo's PhD thesis

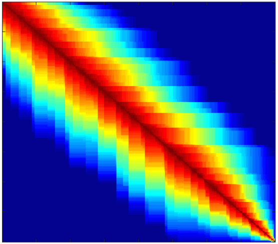
Instances generation



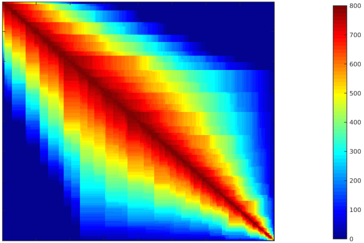
(a) Generation 1



(b) Generation 2



(c) Generation 3



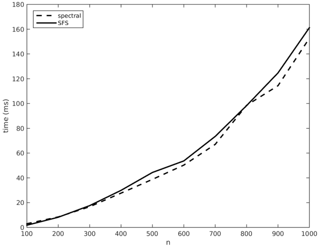
(d) Generation 4

Performance table ($n \leq 1000$)

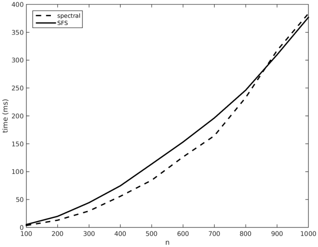
	# distinct values	**low** (≤ 50)			**medium** (> 50 and ≤ 200)			**high** (≥ 200)		
# nonzero entries	**algorithms** / **n**	**spectral**	**SFS**	**LBFS**	**spectral**	**SFS**	**LBFS**	**spectral**	**SFS**	**LBFS**
sparse (≤ 30 %)	**100**	2,98	1,78	10,57	3,68	1,97	58,85	4,24	2,20	-
	200	8,48	8,22	36,99	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 ≤ 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)

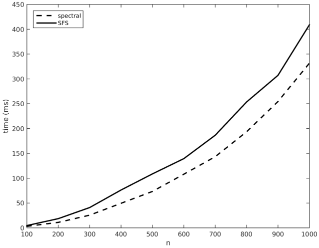
Performance chart ($n \leq 1000$)



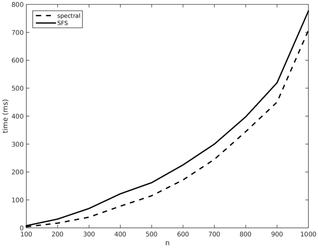
(a) sparse - low



(b) normal - medium



(c) normal - low



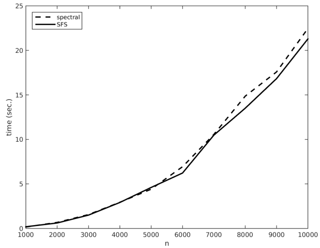
(d) dense - high

Performance table (large instances)

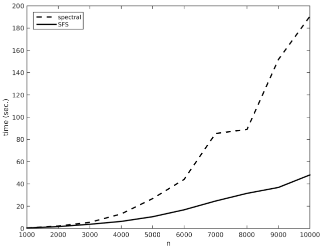
	# distinct values	low (≤ 50)			medium (> 50 and ≤ 200)			high (≥ 200)		
# nonzero entries	algorithms / n	spectral	SFS	LBFS	spectral	SFS	LBFS	spectral	SFS	LBFS
sparse (≤ 30 %)	**1000**	0,16	0,19	-	0,16	0,16	-	0,17	0,18	-
	2000	0,68	0,62	-	0,72	0,7	-	0,76	0,62	-
	3000	1,56	1,5	-	1,95	1,58	-	1,95	1,48	-
	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)

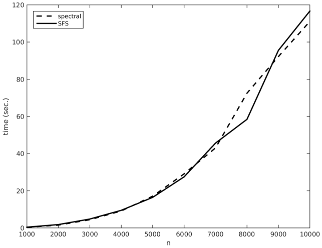
Performance chart (large instances)



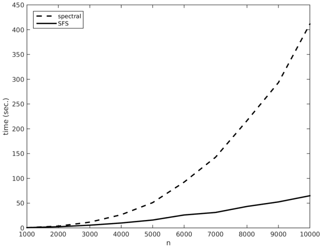
(a) sparse - low



(b) normal - medium



(c) normal - low



(d) dense - high

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- **SFS** (Similarity-First Search) is a new **weighted graph search** algorithm, very simple conceptually and to implement:

  CRAN Package SFS available at the *R* platform.

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- **SFS** (Similarity-First Search) is a new **weighted graph search** algorithm, very simple conceptually and to implement:

  CRAN Package SFS available at the *R* platform.

  SFS permits to recognize Robinsonian matrices.

  Other applications?

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- **SFS** (Similarity-First Search) is a new **weighted graph search** algorithm, very simple 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**.

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- **SFS** (Similarity-First Search) is a new **weighted graph search** algorithm, very simple 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**.

  [L-Tanigawa'17]: Structural characterization for **'chordal' matrices**.

  Other matrix analogues? applications?

Conclusions

- **Lex-BFS** is widely used: recognize chordal graphs (1 sweep, Rose-Tarjan-Lueker'76), unit interval graphs (3 sweeps, Corneil'04), interval graphs (5^* sweeps, Corneil & al.'09), cocomparability graphs (n sweeps, Dusart-Habib'17),...
- **SFS** (Similarity-First Search) is a new **weighted graph search** algorithm, very simple 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**.

  [L-Tanigawa'17]: Structural characterization for **'chordal' matrices**.

  Other matrix analogues? applications?
- ℓ_∞-fitting by Robinsonian is NP-hard to **approximate** within $3/2 - \epsilon$ [Chepoi-Fichet-Seston'09]

  Exists 16-approximation algorithm. [Chepoi-Seston'11]

  Better approximation guarantee?

Thank you

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

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

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

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

M. Laurent, M. Seminaroti, S. Tanigawa.
A Structural Characterization for Certifying Robinsonian Matrices. Electronic Journal of Combinatorics, 2017.

M. Laurent, S. Tanigawa.
Perfect Elimination Orderings for Symmetric Matrices. Opt. Letters'17.