UNIVERSITE MOHAMED BOUDIAF DE M'SILA

Faculté des Mathématiques et de l'Informatique

Département de Mathématiques

MEMOIRE DE FIN D'ETUDE

Présenté pour l'obtention du Diplôme de **MASTER**

Domaine : Mathématiques et Informatique

Filière: Mathématiques

Option : Mathématiques Discrètes

Par

BARKATI *Manel* **ROBEI *Sehila***

Sujet

Algorithmes sur des structures relationnelles

Devant le jury :

Mr. Abdelaziz AMROUNE	Prof. Univ de M'sila	Président
Mr. Lemnaouar ZEDAM	Prof. Univ de M'sila	Rapporteur
Mr. Abdelmadjid BOUDAOUD	Prof. Univ de M'sila	Examinateur
Mr. Souheib MILLES	MA. Univ de M'sila	Examinateur

Promotion : 2016 / 2017

Remerciement

Nous remercions beaucoup Dieu de nous ont aidés pour atteindre ce but de me avoir donner la force et la patience pour accomplir notre travail.

Nous exprimons nos meilleurs remerciement à Monsieur ***Lemnaouar ZADAM*** professeur à l'Université de M'sila, qui a accepté de diriger ce mémoire en témoignant sa confiance.

Nous serait aussi remercions nos vifs remerciement aux membres du jury, Mr ***Abdelaziz AMROUNE, Mr Abdelmadjid BOUDAOUD*** et Mr ***Soheyb MILLES*** d'avoir accepter d'examiner ce travail.

En fin nous remercions tous ceux qui nous ont aidés pour réaliser ce travail.

Algorithmes sur des structures relationnelles

BR *MS*

29 mai 2017

Table des matières

Introduction

La théorie de relation commencé depuis long temps[5], il est originaire sur la théorie des types d'ordres, par HAUSDORFF (Grundzuge der Mengenlehre 1914), SIERPINSKI (Leçons sur les nombres transfinis 1928, taken up again in Cardinal and ordinal numbers 1958), SZPILRAJN (Sur l'extension de l'ordre partiel 1930), DUSHNIK, MILLER (Concerning similarity transformations of 1inearly ordered sets 1940), GLEYZAL (Order types and structure of orders 1940), and to HESSENBERG (Grundbegriffe der Mengenlehre 1906, introducing the negative and rational ordinals).

Les relations entre les éléments d'ensembles se produisent dans de nombreux contextes et Les relations peuvent être utilisées pour modélisé, expliqué et résoudre des problèmes. Elles sont juste des sous-ensembles du produit cartésien de ces ensembles.

Dans ce mémoire, on faites des algorithmes de calculs et de vérifications sur des structures relationnelles. Ces algorithmes sont établir par le Matlab(un programme géométrique fait les opérations d'analyse, la représentation des données, l'intégration, ainsi que la résolution des équations algébrique,....etc).

Ce mémoire est divisé en deux chapitres.

Le premier chapitre est théorique, il est consacré aux relations binaires et leurs propriétés. Ainsi que, des classes particulier des relations binaires. Le deuxième chapitre est pratique, il est contenir des algorithmes de calculs sur les relations et de vérification de ses propriétés. Ces algorithmes sont établir par le Matlab qui est un langage de programmation haute performance, faire les opérations de calculs et d'affichages, avec une simple programmation. Cette langage nous aides pour faire nos algorithmes, surtout les matrices qui nos besoin et prendre long temps pour leur programmation par d'autres langage de programmations.

Chapitre 1

Les relations binaires

Dans ce chapitre, nous donnons quelques notions et définitions de base consernant les relations binaires et leurs propriétés. Ainsi que, des classes particulier des relations binaires. Pour plus des détails, nous nous réferons aux [1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15].

1.1 Les relations binaires

1.1.1 Produit cartésien

Définition 1.1 *[14] Soient E et F deux ensembles non vide, le produit des deux ensembles est le produit cartésien et on note $E \times F$ est défini par :*

$$E \times F = \{(x, y) \mid x \in E, y \in F\}.$$

Exemple 1.1 *Soient E et F deux ensembles finis tel que $E = \{1, 2\}$ et $F = \{a, b, c\}$. Alors,*

(i) *Le produit de $E \times F = \{(1, a), (1, b), (1, c), (2, a), (2, b), (2, c)\}$.*

(ii) *Le produit de $E \times E = \{(1, 1), (1, 2), (2, 1), (2, 2)\}$.*

1.1.2 Relations binaires

Définition 1.2 *[10] Soient $E = \{x_1, x_2, ..., x_n\}$ et $F = \{y_1, y_2, ..., y_m\}$, deux ensembles finis. Un sous-ensemble R de $E \times F$ s'appelle une relation binaire (ou relation, tout simplement) entre E et F, si $(x, y) \in R$ (on dit x est en relation avec y) et on note : xRy.*

***Cas particulier** : Si $E = F$, alors $R \subseteq E \times E$ et on dit que R est une relation sur E.*

Exemple 1.2 *Soient E et F deux ensembles finies tel que $E = \{a, b, c, d\}$ et $F = \{1, 2, 3, 4\}$. Soit R une relation binaire entre E et F définie comme suite :*

$$R = \{(a, 1), (b, 4), (c, 1), (d, 2)\}.$$

1.1.3 Opérations sur les relations binaires

Soient E et F deux ensembles non vides et R, S deux relations entre E et F.

L'inverse d'une relation

L'inverse de la relation $R \subseteq E \times F$ est la relation noté par R^{-1}, définie par :

$$R^{-1} = \{(y, x) \in F \times E \mid (x, y) \in R\}.$$

L'inclusion d'une relation

On dit que $R \subseteq S$ si $(x, y) \in R$ alors $(x, y) \in S$, pour tout $(x, y) \in E \times F$.

L'intersection d'une relation

L'intersection de R et S est la relation $R \cap S$, définie par :

$$R \cap S = \{(x, y) \in E \times F : (x, y) \in R \wedge (x, y) \in S\}.$$

L'union d'une relation

L'union de R et S est la relation $R \cup S$, définie par :

$$R \cup S = \{(x, y) \in E \times F : (x, y) \in R \vee (x, y) \in S\}.$$

Le complément d'une relation

Le complément de la relation R est la relation noté par R^c ou $\overline{R}$, définie par :

$$R^c = \overline{R} = \{(x, y) \in E \times F : (x, y) \notin R\}.$$

La composition d'une relation

Soit δ une relation entre E et T. La composition de R et δ est $R \subseteq E \times F : \delta \subseteq F \times T$, $C = \delta \circ R \subseteq E \times T$ la relation $\delta \circ R$ définie par :

$$\delta \circ R = \{(x,t) \in E \times T \mid (\exists y \in F)((x,y) \in R \text{ et } (y,t) \in \delta)\}.$$

La composition de R et R s'écrit $R \circ R$ ou R^2. R^n est la composition n^{ieme} de R, i.e., $R^n = R \circ R^{n-1}$.

1.2 Représentations d'une relation binaire

Soient E et F deux ensembles finis de cardinal n et m, respectivement soit R une relation entre E et F.
Souvent on peut représenter une relation binaire par une matrice et un graphe (voir, [6, 11])

1.2.1 Représentation matricielle

La représentation de R par une matrice booléenne $M_R = (m_{ij})_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m}$ comportant n lignes et m colonnes. Les éléments de la matrice M_R sont définis par la manière suivant :

$$m_{ij} = \begin{cases} 1, \text{ si } x_i R y_j, \\ 0, \text{ si non .} \end{cases}$$

Exemple 1.3 *La représentation matricielle de la relation de l'exemple* ***1.2*** *est donné par :*

$$M_R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Proposition 1.1 *Si M_R est la matrice associée d'une relation R, la matrice associée de la relation inverse R^{-1} est tout simplement la matrice transposé de M_R, i.e.,*

$$M_{R^{-1}} = (M_R)^t.$$

1.2.2 Représentation graphique

Un graphe est un couple $G = (V, T)$ tel que V est un ensemble des points (parfois dit, sommets ou nœuds) et T est l'ensemble des flèches (parfois dit, arc ou segments) reliés les points de V. Un graphe peut être orienté ou bien dirigé (des flèches) ou non orienté (des traits non-orientées). Toute relations R entre deux ensembles finis E et F peut représentée par un graphe orienté $G_R = (V, T)$ tel que $V = E \cup F$ et T est l'ensemble des flèches qui reliés les points $(x, y) \in R$.

Exemple 1.4 *Soit $E = \{1, 2, 3, 4\}$ et R est la relation sur E donné par :*

$$R = \{(1, 1), (1, 4), (3, 4), (2, 3)\}.$$

Le graphe représentatif de R est :

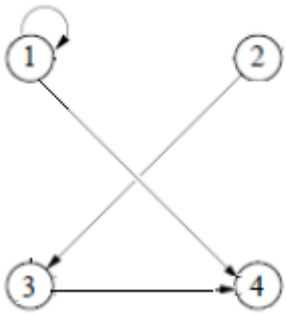


FIGURE 1.1 – Graphe représentatif de R.

1.2.3 Graphe Sagittale

Les éléments des ensembles E et F deviennent des points dans de cercles. La relation R représentée par des arcs orientées ou des flèches relient les points $x_i \in E$ par des points $y_j \in F$ dans le cas ou $(x_i, y_j) \in R$.

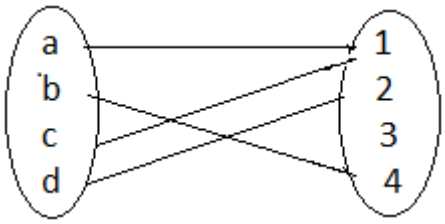


FIGURE 1.2 – Graphe Saggittale de l'exemple **1.2**.

1.3 Propriétés des relations binaires

Dans cette section, nous nous présentons des propriétés d'une relation binaire définit sur un seul ensemble.

1.3.1 Réflexivité

Une relation binaire R sur un ensemble E est dite **réflexive** si pour tout $x \in E$, alors $(x, x) \in R$.

Exemple 1.5 *Soit $E = \{1, 2, 3, 4, 5, 6\}$ et R est la relation sur E définit par :*

$$xRy \Leftrightarrow x \text{ divise } y.$$

i.e., $R = \{(1,1), (1,2), (1,3), (1,4), (1,5), (1,6), (2,2), (2,4), (2,6), (3,3), (3,6), (4,4), (5,5), (6,6)\}$. Le fait que $\{(1,1), (2,2), (3,3), (4,4), (5,5), (6,6)\} \subset R$, implique alors que R est réflexive.

Remarque 1.1

- *La matrice associe d'une relation réflexive est une matrice de diagonal 1.*
- *Tout sommet dans le graphe représentatif de R, il y a un boucle (auto-flèche).*

1.3.2 Irréflexivité

Une relation binaire R sur un ensemble E est dite **irréflexive** si pour tout $x \in E$, alors $(x, x) \notin R$.

Exemple 1.6 *Soit $E = \{1, 2, 3, 4, 5, 6\}$ et R est la relation sur l'ensemble E définit par :*

$$R = \{(1,2), (2,4), (4,1), (6,3)\}.$$

Le fait que $\{(1,1),(2,2),(3,3),(4,4),(5,5),(6,6)\} \nsubseteq R$*, implique alors que* R *est irréflexive.*

Remarque 1.2

- *La matrice associe d'une relation irréflexive est une matrice de diagonale* 0*,*
  i.e., $(m_{ii} = 0 \forall\ 1 \leq i \leq n)$*.*
- *Le graphe d'une relation irréflexive , il n 'y 'a pas un boucle de sommet.*

1.3.3 Non-réflexivité

Une relation R sur E est dite **non-réflexive** s'il existe $x \in E$ tel que $(x,x) \notin R$.

Exemple 1.7 *Soit* $E = \{1,2,3,4,5\}$ *et* R *est la relation sur* E *donné par :*

$$R = \{(1,1),(2,2),(3,2),(5,4)\}.$$

Le fait que $(3,3),(4,4)$ *et* $(5,5) \notin R$*, implique que* R *est non-réflexive.*

Remarque 1.3

- *Le diagonale d'une matrice associé d'une relation non-réflexive contient au moins un* 0*.*
- *Le graphe d'une relation non-réflexive non contient au moins un boucle de sommet.*

1.3.4 Symétrie

Une relation R sur E est dite **symétrique** si $(x,y) \in R$, alors $(y,x) \in R$, i.e., xRy et yRx.

Exemple 1.8 *Soit* $E = \{1,2,3,4\}$ *et* R *est la relation sur* E *définit par :*

$$R = \{(1,1),(1,4),(2,2),(2,3),(3,2),(3,4),(4,1),(4,3)\}.$$

Le fait que $(x,y) \in R$ *implique que* $(y,x) \in R$*, alors* R *est symétrique.*

Remarque 1.4

- *La matrice associe d'une relation symétrique est une matrice symétrique.*
- *Dans le graphe d'une relation symétrique, tout couple de sommets, il y 'a un arc-aller et un arc-retour.*

1.3.5 Anti-symétrie (faiblement anti-symétrie)

Une relation R sur E est dite **anti-symétrique** si $(x, y) \in R$ et $(y, x) \in R$, alors $x = y$, i.e., $\forall x, y \in E, (xRy)$ et $(yRx) \Longrightarrow x = y$.

Exemple 1.9 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4, 5, 6\}$, et R est la relation sur E donné par :*

$$R = \{(1, 2), (2, 2), (2, 4), (2, 5), (4, 1), (5, 4), (6, 3)\}.$$

Le fait que $(x, y) \in R$, implique que $(y, x) \notin R$, alors R est anti-symétrique.

Remarque 1.5

- *Dans la matrice associe d'une relation binaire anti-symétrique on a : $m_{ij} = m_{ji} = 1 \Longrightarrow i = j$.*
- *Dans le graphe d'une relation anti-symétrique, s'il existe un arc-aller entre deux sommets déférent, alors il n'y a pas une arc-retour .*

1.3.6 Asymétrie (fortement anti-symétrie)

Une relation R sur E est dite **asymétrique** si $(x, y) \in R$, alors $(y, x) \notin R$.

Exemple 1.10 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4, 5, 6\}$, et R est la relation sur E donné par :*

$$R = \{(1, 2), (2, 4), (2, 5), (5, 4), (6, 3)\}.$$

Le fait que $(x, y) \in R$ implique que $(y, x) \notin R$, alors R est asymétrique.

Proposition 1.2 *[6] Toute relation asymétrique est irréflexive.*

Preuve 1.1 *Soit R une relation asymétrique, et **a** un point tel que aRa, par l'asymétrie de R, on aurait $aRa \Longrightarrow \neg\, aRa$, et aRa serait contradictoire, il n'existe donc aucun point pour lequel on ait aRa. Donc R est irréflexive.*

Remarque 1.6

- *Dans la matrice associe d'une relation binaire asymetrique si : $m_{ij} = 1 \Longrightarrow m_{ji} = 0$ et $m_{ii} = 0$.*
- *Dans le graphe d'une relation asymétrique, s'il existe un arc-aller entre deux sommets, alors il n'y 'a pas une arc-retour et il n 'y 'a pas un boucle de sommet.*

1.3.7 Transitivité

Une relation R sur E est dite **transitive** si pour tout $x, y, z \in E$ on a $(x, y) \in R$ et $(y, z) \in R$, alors $(x, z) \in R$, i.e.,

$$\forall\, x, y, z \in E, ((x, y) \in R \wedge (y, z) \in R) \Longrightarrow (x, z) \in R.$$

Exemple 1.11 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4\}$ et R est la relation sur E donné par :*

$$R = \{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\}.$$

Le fait que $\forall\, x, y, z \in E$, on a :

$$\textit{si} \begin{cases} (x, y) \in R \\ \textit{et} \\ (y, z) \in R \end{cases} \textit{, alors } (x, z) \in R, \textit{ implique que } R \textit{ est transitive.}$$

1.3.8 Anti-transitivité :(Itransitivité)

Une relation R sur E est dite **anti-transitive** si pour tout $x, y, z \in E$ tel que $(x, y) \in R$ et $(y, z) \in R$, alors $(x, z) \notin R$, i.e.,

$$\forall\, x, y, z \in E, (xRy) \wedge (yRz) \Longrightarrow (x, z) \in R^c.$$

Exemple 1.12 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4, 5, 6\}$ et R est la relation sur E donné par :*

$$R = \{(1, 2), (2, 5), (4, 1), (4, 5), (5, 4), (6, 3)\}.$$

Le fait que $\forall\, x, y, z \in E$, on a :

$$\textit{si} \begin{cases} (x, y) \in R \\ \textit{et} \\ (y, z) \in R \end{cases} \textit{, alors } (x, z) \notin R, \textit{ implique que } R \textit{ est anti-transitive.}$$

Proposition 1.3 *[6] Toute relation anti-transitive est irréflexive.*

Preuve 1.2 *Soit R une relation sur un ensemble E anti-transitive. Soit $\boldsymbol{a} \in E$ tel que aRa. Par l'anti-transitivité de R, on aurait que $aRa \wedge aRa \Longrightarrow aR^ca$. aRa et aR^ca serait contradictoire. Alors il n'existe aucun point pour lequel on ait aRa. Donc R est irréflexive.*

1.3.9 Circularité

Une relation R sur E est dite **circulaire** si pour tout $x, y, z \in E$ tel que $(x, y) \in R$ et $(y, z) \in R$, alors $(z, x) \in R$, i.e.,

$$\forall\, x, y, z \in E, (xRy) \wedge (yRz) \Longrightarrow (z, x) \in R.$$

Exemple 1.13 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4, 5, 6\}$, et R est la relation sur E donné par :*

$$R = \{(1, 3), (2, 4), (2, 6), (4, 5), (5, 2), (6, 5)\}.$$

Le fait que $\forall\, x, y, z \in E$, on a :

$$\text{si} \begin{cases} (x, y) \in R \\ \text{et} \\ (y, z) \in R \end{cases} \text{, alors } (z, x) \in R, \text{ implique que } R \text{ est circulaire.}$$

1.4 Classes particuliers des relations binaires

1.4.1 Relation d'équivalence

Définition 1.3 *Soit R une relation sur E. R est dite relation d'équivalence si elle est **réflexive**, **symétrique** et **transitive**. Une relation d'équivalence est noté généralement par $\approx$.*

Exemple 1.14 *Soient $E = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ et R est la relation d'équivalence sur E donnée par :*

$$\begin{aligned} R = \{&(1, 1), (1, 2), (1, 3), (2, 1), (2, 2), (2, 3), (3, 1), (3, 2), \\ &(3, 3), (4, 4), (4, 5), (5, 4), (5, 5), (6, 6), (6, 7), (6, 8), \\ &(6, 9), (7, 6), (7, 7), (7, 8), (7, 9), (8, 6), (8, 7), (8, 8), \\ &(8, 9), (9, 6), (9, 7), (9, 8), (9, 9)\}. \end{aligned}$$

Alors, R est une relation d'équivalence.

Exemple 1.15 *Soit E un ensemble non vide, alors l'égalité est toujours une relation d'équivalence sur E.*

Définition 1.4 *: Soit $\approx$ une relation d'équivalence sur E.*

(i) On dit que deux éléments $x, y \in E$ sont équivalents si $x \approx y$.

(ii) On appelle classe d'équivalence d'un élément $x \in E$, l'ensemble $\overline{x} = [x] = \{y \in E \mid x \approx y\}$.

(iii) x est dit un représentant de la classe d'équivalence $\overline{x}$.

(iv) On appelle ensemble quotient de E par la relation d'équivalence $\approx$, l'ensemble des classes d'équivalences de tous les éléments de E. Cet ensemble est noté $E/_{\approx}$, tel que :

$$E/_{\approx} = \{\overline{x} \mid x \in E\}.$$

Exemple 1.16 *Dans $\mathbb{R}$ on définit la relation R par :*

$$(\forall x, y \in R)(xRy \Longleftrightarrow x^2 - 1 = y^2 - 1).$$

Alors, R est une relation d'équivalence. En effet,

(i) R est une relation réflexive, car $(\forall x \in \mathbb{R})(x^2 - 1 = x^2 - 1)$.

(ii) Soit $x, y \in \mathbb{R}$ tel que xRy. Alors $x^2 - 1 = y^2 - 1$. Donc $y^2 - 1 = x^2 - 1$.
D'où, yRx. En conséquent R est symétrique.

(iii) Soit $x, y, z \in \mathbb{R}$ tel que xRy et yRz. Alors, $(x^2 - 1 = y^2 - 1) \wedge (y^2 - 1 = z^2 - 1)$.
Ce qui implique $(x^2 - 1 = z^2 - 1)$.
Donc, xRz. D'où, R est transitive.

En conséquent, R est une relation d'équivalence.

L'ensemble quotient $\mathbb{R}/R$ est :
Soit $x, y \in \mathbb{R}$

$$\begin{aligned}
\mathbb{R}/R = \{\overline{x} \mid\ x \in \mathbb{R}\} \text{ , tel que } \overline{x} &= \{y \in \mathbb{R} \mid yRx\} \\
&= \{y \in \mathbb{R} \mid y^2 - 1 = x^2 - 1\} \\
&= \{y \in \mathbb{R} \mid y^2 - x^2 = 0\} \\
&= \{y \in \mathbb{R} \mid (y - x)(y + x) = 0\} \\
&= \{y \in \mathbb{R} \mid y = x \text{ ou } y = -x\} \\
&= \{x,\ -x \mid x \in \mathbb{R}\}.
\end{aligned}$$

Donc $\mathbb{R}/R = \{\{x,\ -x\} \mid x \in \mathbb{R}\}$.

Théorème 1.1 *[8] Les classes d'équivalence d'une relation d'équivalence sur un ensemble E sont les forme une partition de E.*

1.4.2 Relation d'ordre

Définition 1.5 *La relation R dans E est dite une relation d'ordre si elle est réflexive, anti-symétrique et transitive. La relation d'ordre noté généralement par $\leqslant$.*
Un ensemble E muni d'une relation d'ordre est dit un ensemble ordonné, on note par $(E, \leqslant)$.

Diagramme de Hasse d'une relation d'ordre : On définit maintenant le diagramme de Hasse pour un ensemble fini partiellement ordonné. A chaque élément de E on associe un point du plan, et on trace une ligne de x à y si $x < y$. On veille à ce que ces lignes n'intersectent pas les autres points, et on veille à ce que $x < y$ implique que l 'ordonnée du point associé à x soit inférieure à l 'ordonnée du point associé à y [1].

Exemple 1.17 *Soit $D(30)$ l'ensemble des diviseurs positives de 30, i.e., $D(30) = \{1, 2, 3, 5, 6, 10, 15, 30\}$ ordonné par la relation divisibilité.*

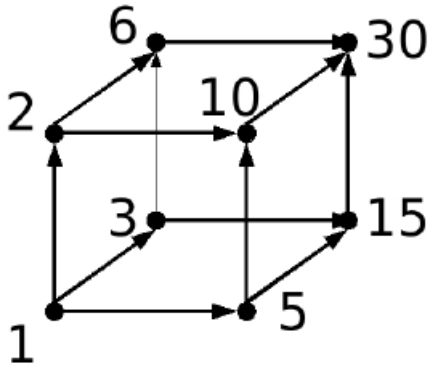


FIGURE 1.3 – Diagramme de Hasse de $(D(30), |)$

Exemple 1.18 *Soit $\mathbb{R}$ l'ensemble des réels, les relations $(\leq, \geq)$ son des relations d'ordres.*

Exemple 1.19 *Soit R est la relation définie sur $\mathbb{N}^*$ par :*

$$pRq \iff (\exists n \in \mathbb{N}^* \text{ tel que } p^n = q).$$

Alors, R est une relation d'ordre. En effet,

(i) La réflexivité :

on a : $p^1 = p \Longrightarrow pRp$. *Donc,* R *est réflexive.*

(ii) L'antisymétrie : soit $p, q \in \mathbb{N}^*$ *tel que* pRq *et* qRp. *Alors,*

$$\begin{cases} \exists\, n_1 \in \mathbb{N}^* \text{ tel que } p^{n_1} = q \\ \text{et} \\ \exists\, n_2 \in \mathbb{N}^* \text{ tel que } q^{n_2} = p. \end{cases}$$

Donc, $(p^{n_1 n_2} = p)$. *Ce qui implique que* $n_1 n_2 = 1$, *et donc* $n_1 = n_2 = 1$. *D'où,* $p = q$.
En conséquence, R *est antisymétrique.*

(iii) La transitivité : soit $p, q, r \in \mathbb{N}^*$ *tel que* pRq *et* qRr. *Alors,*

$$\begin{cases} \exists n_1 \in \mathbb{N}^* \text{ telque } p^{n_1} = q \\ \text{et} \\ \exists n_2 \in \mathbb{N}^* \text{ telque } q^{n_2} = r. \end{cases}$$

Donc $p^{n_1 n_2} = r$, *Ce qui implique que* $(\exists m = n_1 n_2 \in \mathbb{N}^*$ *tel que* $p^m = r)$, *i.e.,* pRr. *D'où* R *est transitive.*

En conséquence (i), (ii) *et* (iii) *déduire que* R *est* ***une relation d'ordre****.*

Définition 1.6 ***(Ordre total)****[13] Soit* R *une relation d'ordre sur un ensemble* E. *On dit que* R *est total (ou linéaire) si pour tous* $x, y \in E$, *alors* xRy *ou* yRx. *Autrement dit, chaque deux éléments* x, y *sont comparables. Dans ce cas* (E, R) *est dite une chaîne. Dans le cas contraire, on dit que* R *est un ordre partiel.*

Définition 1.7 ***(Ordre strict)*** *Soit* R *une relation sur* E. R *est dit un ordre strict s'il est* ***irréflexive*** *et* ***transitive****. Un ensemble muni d'un ordre strict est dit un ensemble strictement ordonné, on le note par* $(E, <)$.

Exemple 1.20 *Dans* $\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}$ *la relation* $<$ *est l'ordre strict linéaire associé à* $\leq$. *Par contraire,* $\subset$ *est un ordre strict (non linéaire).*

Exemple 1.21 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4, 5\}$ et R est la relation sur E donné par :*

$$R = \{(1,3), (1,4), (1,5), (2,1), (2,3), (2,4), (2,5), (3,4), (3,5), (4,5)\}.$$

Alors, R est un ordre strict sur E.

Définition 1.8 ***(Préordre)*** *Soit R une relation sur E. R est dite un Préordre sur E si elle est **réflexive** et **transitive**. N'est pas symétrique et n'est pas antisymétrique en général.*

Exemple 1.22 *Soit E un ensemble fini tel que $E = \{1, 2, 3, 4\}$ et R est la relation sur E donné par :*
$R = \{(1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (2,4),$
$(3,2), (3,3), (3,4), (4,2), (4,3), (4,4)\}$. *Alors, R est une relation préordre.*

Définition 1.9 ***(antichaîne)*** *L'antichaîne est un ensemble ordonné dans lequel chaque deux éléments distincts sont toujours incomparables, i.e., $(x \not\leqslant y$ et $y \not\leqslant x)$.*

Ordre induit et ordre produit

Définition 1.10 ***(Ordre induit)*** *Soit R une relation d'ordre sur E et F une partie de E, la restriction à F de la relation R est une relation d'ordre sur F, qui est appelé l'ordre induit par R dans F.*

Définition 1.11 ***(Ordre produit)*** *Soient (E, R) et (F, S) sont deux ensembles ordonnés par les relations notées R et S peut ordonner le produit cartésien $E \times F$ par l'ordre T, dit ordre produit de R et S donné par :*

$$(a, b)T(c, d) \Longleftrightarrow (aRc \text{ et } bSd).$$

1.4.3 Élément particuliers d'un ensemble ordonné

Définition 1.12 ***(Majorant/minorant)*** *Soit $\leq$ une relation d'ordre sur un ensemble E et A une partie de E. Alors,*

*(i) On dit que A est **majorée** s'il existe $M \in E$ tel que $(\forall a \in A)(a \leq M)$. Un tel M est appelé un majorant de A.*

*(ii) On dit que A est **minorée** s'il existe $m \in E$ tel que $(\forall a \in A)(m \leq a)$. Un tel m est appelé un minorant de A.*

*(iii) On dit que A est **bornée** si elle est à la fois majorée et minorée.*

Définition 1.13 ***(Plus grand élément (maximum)/ Plus petit élément (minimum))*** *Soient $\leq$ une relation d'ordre sur un ensemble E et A une partie de E. Alors,*

*(i) On appelle **plus grand élément** de A ou maximum de A tout élément de A qui majore A. Un tel plus grand élément est unique et donc appelé le plus grand élément de A, noté $maxA$.*

*(ii) On appelle **plus petit élément** de A ou minimum de A tout élément de A qui minore A. Un tel plus petit élément est unique et donc appelé le plus petit élément de A, noté $minA$. Un tel m est appelé un minorant de A.*

*(iii) On dit que A est **bornée** si elle est à la fois majorée et minorée.*

Définition 1.14 ***(Borne supérieure/ Borne inférieure)*** *Soit $\leq$ une relation d'ordre sur un ensemble E et A une partie de E. Alors,*

(i) Le plus petit majorant de A (s'il exist) est appelé la borne supérieure de A et noté $supA$.

(ii) Le plus petit minorant de A (s'il exist) est appelé la borne inférieure de A et noté $infA$.

1.4.4 Nombres des relations binaires et ses classes sur des ensembles finis

Considérons un ensemble E fini de cardinal n et un ensemble F fini de cardinal p. Il y a autant des relations binaires de E sur F que des applications de $E \times F$ dans $\{0, 1\}$. Ce qui donne 2^{np} relations.
En particulier, si $E = F$, on trouve 2^{n^2} relations binaires sur E. On sait aussi :

- $2^{n(n-1)}$ relations réflexives.
- $2^{n(n+1)/2}$ relations symétriques.
- $2^{n(n-1)/2}$ relations réflexives et symétriques.
- Pour le nombre de relations transitives, on n'a pas une formule fermée.

Chapitre 2

Algorithmes sur les relations binaires

Dans ce chapitre nous donnons quelques algorithmes sur les relations binaires et ses propriétés. Les algorithmes sont établir par le Matlab.

2.1 Algorithmes

Une relation binaire dans un ensemble fini est représenté généralement par une matrice (ou bien une table) binaire. Cette matrice est utiliser pour établir des algorithmes de calculs sur les relations et de vérification de ses propriétés.

Définition 2.1 (Algorithme) *Un algorithme est une suite d'étapes dont le but est de résoudre un problème ou d'accomplir une tâche.*
Selon le dictionnaire Larousse : Un algorithme est un ensemble des règles opératoires dont l'application permet de résoudre un problème énoncé au moyen d'un nombre fini d'opérations. Un algorithme puet être traduit, grâce à un langage de programmation, en un programme exécutable par un ordinateur [16].

2.2 Algorithmes sur les opérations entre les relations

Dans cette section, on va donner quelques algorithmes avec des exemples d'exécution sur les opérations de bases sur les relations binaires.

2.2.1 L'inverse et le complément d'un relation binaire

Algorithme qui calcule l'inverse et le complément d'une relation binaire.

function[ ] = invcomp(R)

clc

$[m, n]$ = size(R);

% **Invers**.....

Invers = zeros(n);

for $i = 1 : n$

for $j = 1 : n$

Invers$(i, j) = R(j, i)$;

end

end

Invers

% **le complement**.....

Complement = ones$(n) - R$

end

Exemple 2.1

```
R =

     1     0     0
     0     0     1
     1     1     1

>> invcomp(R)

Invers =

     1     0     1
     0     0     1
     0     1     1


Complement =

     0     1     1
     1     1     0
     0     0     0
```

2.2.2 L'union et l'intersection et la composition des relations binaires

L'algorithme suivant calcule l'union, l'intersection et la composition des relations binaires.

function[ ] = unintercomps(R, S)

clc;

$[m, n]$ = size(R);

% **l'union.....**

Union=max(R, S)

% **l'intersection.....**

Intersection=min(R, S)

% **La composition.....**

v=zeros$(1, n)$;

Composition = zeros$(1, n)$;

for $i = 1 : n$

for $j = 1 : n$

for $k = 1 : n$

v(k) = min$(R(i, k), S(k, j))$;

end

Composition(i, j) = max(v);

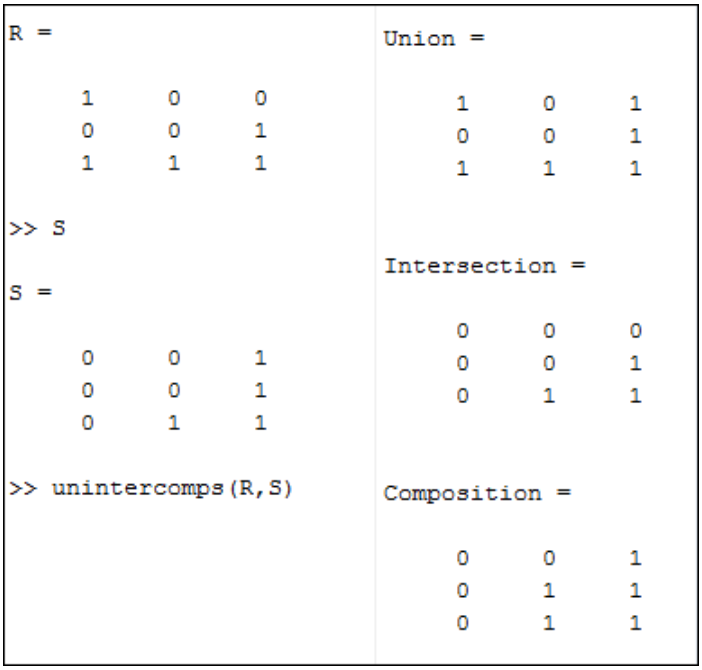
end

end

Composition

end

Exemple 2.2



2.3 Algorithmes sur les propriétés des relations binaires

Dans cette section, nous nous donnons quelques algorithmes avec des exemples d'exécution sur les propriétés des relations binaires.

2.3.1 Réflexivité

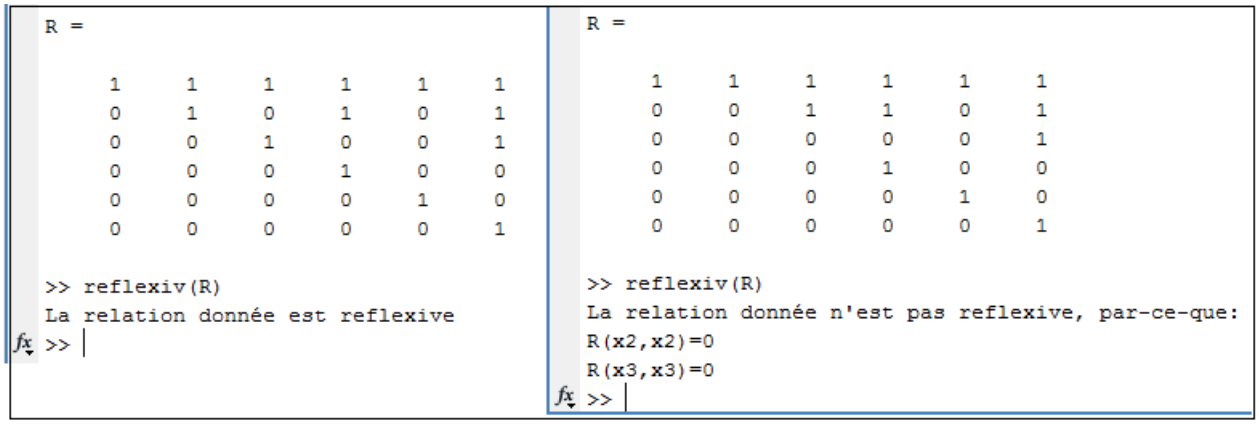
L'algorithme suivant vérifie la réflexivité d'une relation binaire.

function[ ] = reflexiv(R)
clc; $[m, n]$ = size(R);
if (diag(R) == ones(n, 1))
fprintf ('La relation R est reflexive\n')
else
fprintf('La relation R n"est pas reflexive, par-ce-que : \n')
for $i = 1 : m$
if $R(i, i) == 0$
fprintf('(x% d, x% d) = 0 \ n', i, i)
end
end

end

end

Exemple 2.3



2.3.2 Irréflexivité

L'algorithme suivant vérifie l'irréflexivité d'une relation binaire.

function[ ] = irreflexive(R)

clc; $[m, n]$ = size(R);

if (diag(R) == zeros($n, 1$))

fprintf ('La relation donnée est irreflexive\n')

else

fprintf('La relation donnée n"est pas irreflexive, par-ce-que : \n')

for $i = 1 : m$

if $R(i, i) \sim= 1$

fprintf('$(x\%\ d, x\%\ d) = 1 \setminus$ n', i, i)

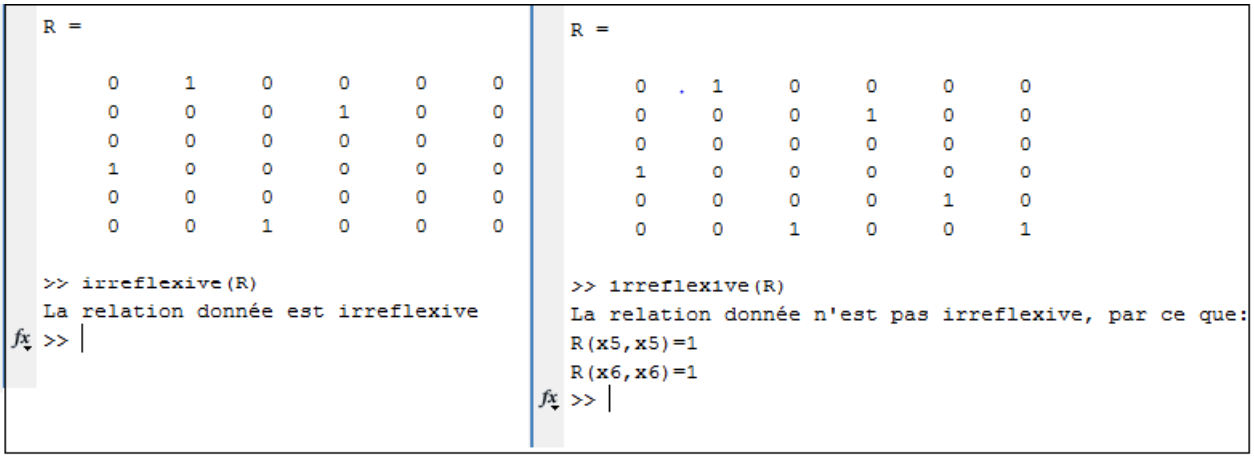
end

end

end

end

Exemple 2.4



2.3.3 Non-réflexivité

L'algorithme suivant vérifie la relation non-réflexivité d'une relation binaire.

function[ ] = nonreflexive(R)

clc; $[m, n]$ = size(R);

if (diag(R) == zeros($n, 1$))

fprintf ('La relation R n"est pas non reflexive, par-ce-qu"elle irréflexive.\n')

elseif (diag(R) == ones($n, 1$))

fprintf('La relation R n"est pas non reflexive, par-ce-qu"elle réflexive.\n')

else

fprintf('La relation R est non reflexive. \n')

end

Exemple 2.5

```
R =

     0     1     0     0     0     0
     0     0     0     1     0     0
     0     0     0     0     0     0
     1     0     0     0     0     0
     0     0     0     0     1     0
     0     0     1     0     0     1

>> nonreflexive(R)

ans =

La relation R est non réflexive
```

```
R =

     0     1     0     0     0     0
     0     0     0     1     0     0
     0     0     0     0     0     0
     1     0     0     0     0     0
     0     0     0     0     0     0
     0     0     1     0     0     0

>> nonreflexive(R)

ans =

La relation R n'est pas non réflexive, par-ce-qu'elle irréflexive.
```

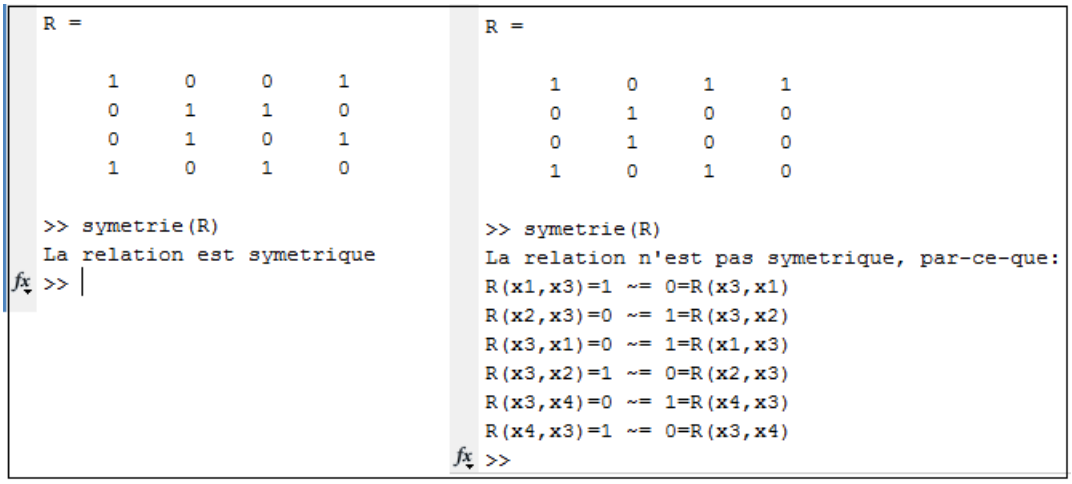
2.3.4 Symétrie

L'algorithme suivant vérifie la symétrie d'une relation binaire.

```
function[ ] = symetrie(R)
clc; [m, n] = size(R); Sym = 1;
for i = 1 : m
    for j = 1 : n
        if (i ~= j && R(i, j) ~= R(j, i))
            Sym = 0;
        end
    end
end
if (Sym ~= 0)
    fprintf('La relation R est symétrique. \n')
 else
    fprintf('La relation R n"est pas symétrique, par-ce-que : \n')
        for i = 1 : m
            for i = 1 : n
                if (i ~= j && R(i, j) ~= R(j, i))
    fprintf('R(x %d, x % d) = % d ~= % d = R(x % d, x % d)', i, j, R(i, j), R(j, i), j, i)
                end
            end
```

```
        end
    end
end
```

Exemple 2.6 *La notation du le symbole $\neq$ dans le Matlab est noté par $\sim=$.*



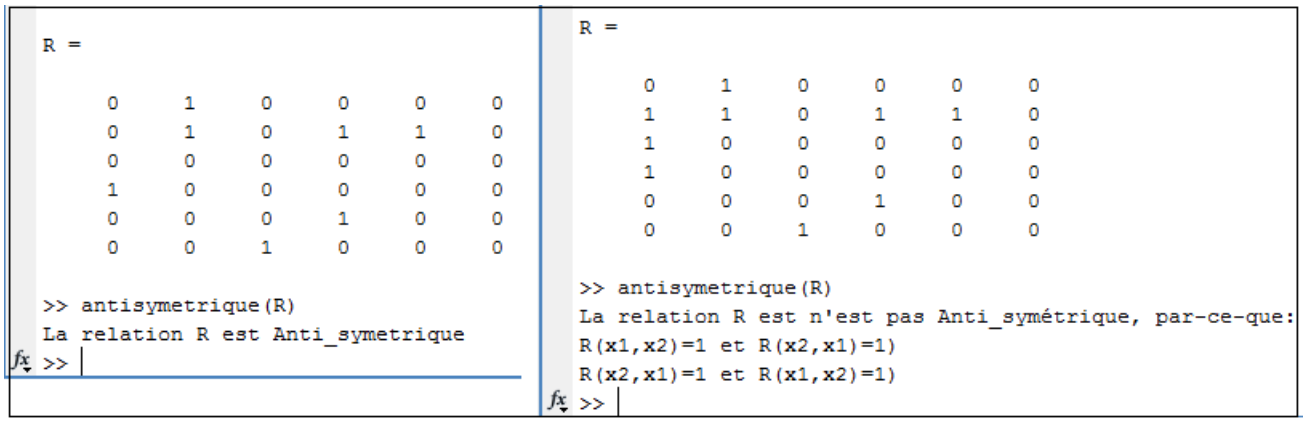
2.3.5 Anti-symétrie

L'algorithme suivant vérifie l'anti-symétrie d'une relation binaire.

```
function[ ] = antisymetrique(R)
clc; [m, n] = size(R); Ansym = 1;
for i = 1 : m
    for j = 1 : n
        if (i ~= j && R(i, j) == 1 && R(j, i) == 1)
            Ansym = 0;
        end
    end
end
if (Ansym == 1)
    fprintf('La relation R est antisymétrique. \n')
```

```
else
    fprintf('La relation R n''est pas antisymétrique, par-ce-que : \n')
    for i = 1 : m
        for j = 1 : n
            if (i ~= j && R(i,j) == 1 && R(j,i) == 1)
                fprintf('R(x %% d,x %% d) = 1 et R(x %% d,x %% d) = 1 \n',i,j,j,i)
            end
        end
    end
end
end
```

Exemple 2.7



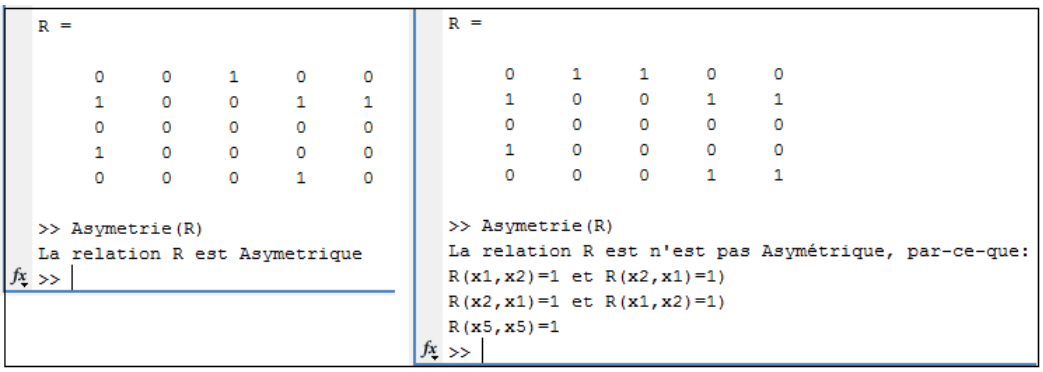
2.3.6 Asymétrie

L'algorithme suivant vérifie la asymétrie d'une relation binaire.

```
function[ ] = asymetrie(R)
clc; [m,n] = size(R); Asym = 1;
for i = 1 : m
    for j = 1 : n
```

```
        if (i ~= j && R(i,j) == 1 && R(j,i) == 1) || (R(i,i) == 1)
            Asym = 0;
        end
    end
end
if (Asym == 1)
    fprintf('La relation R est asymétrique \n')
else
    fprintf('La relation R n"est pas asymétrique, par-ce-que : \n')
    for i = 1 : m
        for j = 1 : n
            if (i ~= j && R(i,j) == 1 && R(j,i) == 1 || R(i,i) == 1)
                if (i ~= j && R(i,j) == 1 && R(j,i) == 1)
                elseif (i == j) && (R(i,j) == 1)
                    fprintf('R(x % d, x % d) = 1 \ n', i, j)
                end
            end
        end
    end
end
end
```

Exemple 2.8

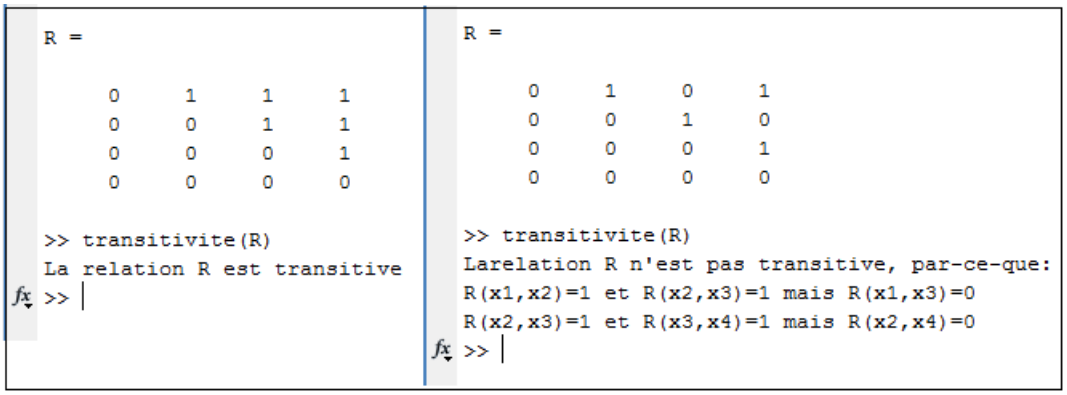


2.3.7 Transitivité

L'algorithme suivant vérifie la transitivité d'une relation binaire.

```
function[ ] = transitivite(R)
clc; Testtrans = 1; [m, n] = size(R);
for i = 1 : m
    for j = 1 : n
        for k = n : 1
            if ((R(i, k) == 1) && (R(k, j) == 1) && (R(i, j) == 0))
                Testtrans = 0;
            end
        end
    end
end
if Testtrans == 1
    fprintf('La relation R est transitive. \n')
else
    fprintf('La relation R n"est pas transitive, par-ce-que : \n')
for i = 1 : m
    for j = 1 : n
        for k = n : 1
            if (R(i, k) == 1) && (R(k, j) == 1) && (R(i, j) == 0)
fprintf('R(x % d, x % d) = % d , R(x % d, x % d) = % d et R(x % d, x % d) = % d \ n'
, i, k, R(i, k), k, j, R(k, j), i, j, R(i, j))
            end
        end
    end
end
end
end
```

Exemple 2.9



2.3.8 Anti-transitivité

L'algorithme suivant vérifie l'anti-transitivité d'une relation binaire.

function[ ] = Antitransitve(R)

clc; Testantitrans $= 1$; $[m, n] =$ size(R);

for $i = 1 :\ m$

  for $j = 1 :\ n$

    for $k = 1 :\ n$

      if $((R(i,k) == 1)\ \&\&\ (R(k,j) == 1)\ \&\&\ (R(i,j) == 1))$

        Testantitrans $= 0$;

      end

    end

  end

end

if Testantitrans $== 1$

  fprintf('La relation R est antitransitive. \n')

else

  fprintf('La relation R n"est pas antitransitive, par-ce-que : \n')

for $i = 1 : m$

for $j = 1 : n$

for $k = 1 : n$

if $((mR(i,k) == 1)\ \&\&\ (mR(k,j) == 1)\ \&\&\ (mR(i,j) == 1))$

fprintf$('R(x\ \%\ d, x\ \%\ d) = \%\ d\ ,\ R(x\ \%\ d, x\ \%\ d) = \%d$ et $R(x\ \%\ d, x\ \%\ d) = \%\ d\ \backslash n'$

$, i, k, R(i,k), k, j, R(k,j), i, j, R(i,j))$

end

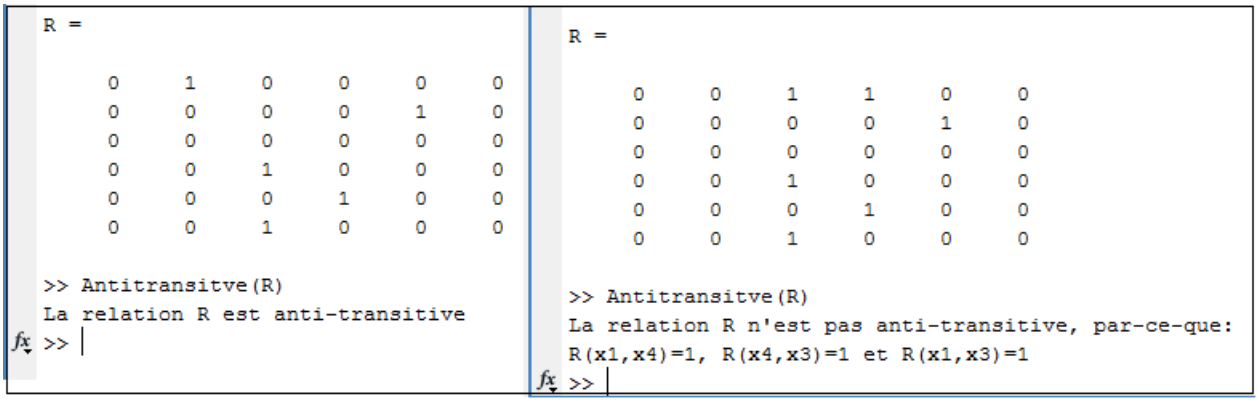
end

end

end

end

Exemple 2.10



2.3.9 Circularité

L'algorithme suivant vérifie si une relation binaire est circulaire ou non.

function$[\ ]$ = circulaire(R)

clc; Test $= 1$; $[m,n] =$ size(R);

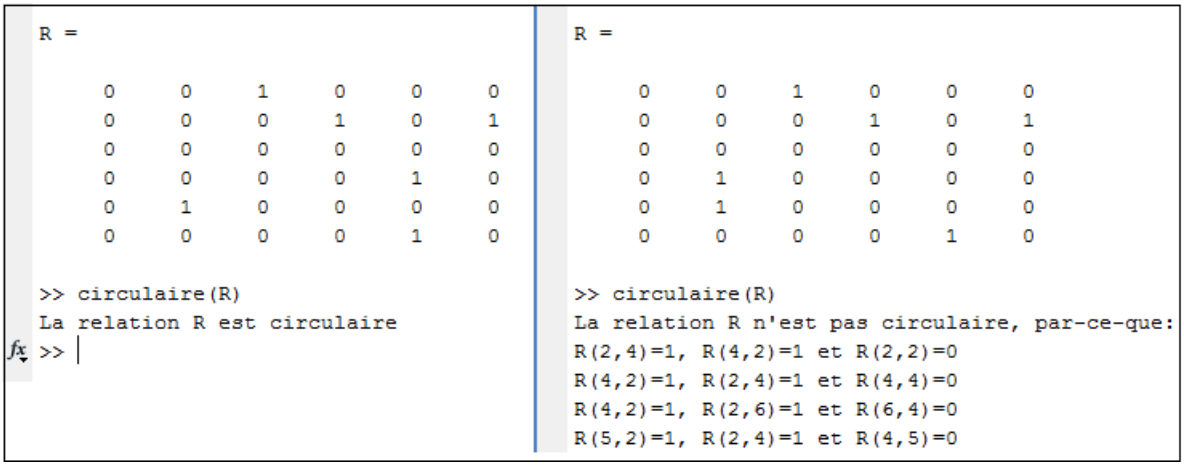
for $i = 1 : n$

for $j = 1 : n$

for $k = 1 : n$

```
            if((R(i,k) == 1) && (R(k,j) == 1) && (R(j,i) == 0))
                Test = 0;
            end
        end
    end
end
if Test == 1
    fprintf('La relation R est circulaire. \n')
else
    fprintf('La relation R n"est pas circulaire, par-ce-que : \n')
for i = 1 : m
    for j = 1 : n
        for k = 1 : n
            if ((R(i,k) == 1) && (R(k,j) == 1) && (R(j,i) == 0))
fprintf('R(%d,%d) = %d , R(%d,%d) = %d et R(%d,%d) = %d \n'
,i,k,R(i,k),k,j,R(k,j),j,i,R(j,i))
            end
        end
    end
end
end
end
```

Exemple 2.11



2.4 Algorithmes sur des classes particulières des relations binaires

Dans cette section, on va donner quelques algorithmes avec des exemples d'exécutions sur des classes particulières des relations binaires.

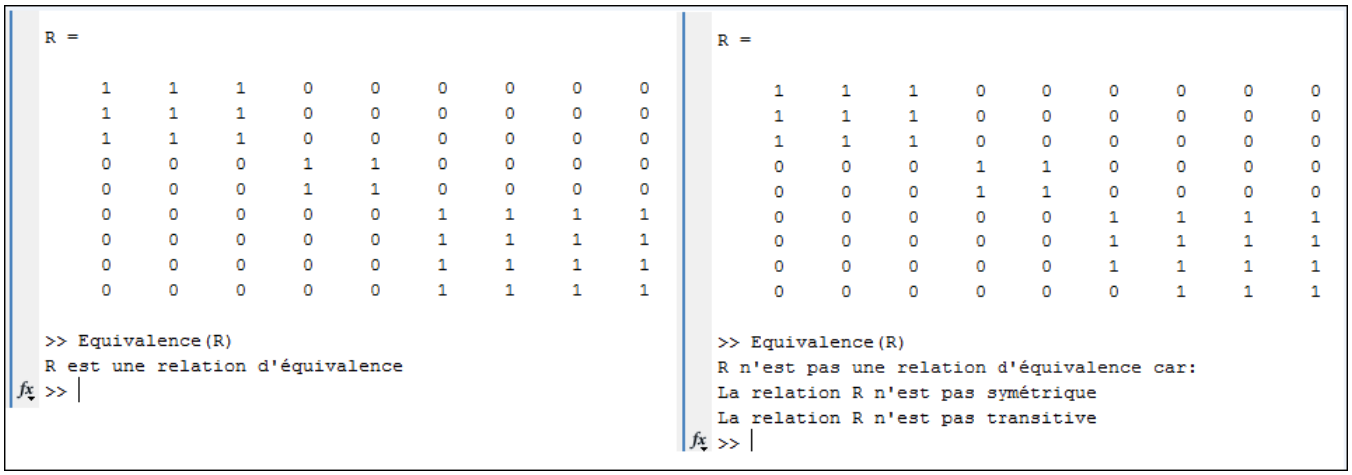
2.4.1 Relation d'équivalence

L'algorithme suivant vérifie que si une relation donnée est une relation d'équivalence ou non.

function[ ] = Equivalence(R)
clc; ref = 0; Sym = 1; Testtrans = 1; $[m, n]$ = size(R);
if (diag(R) == ones(n, 1))
    ref = 1;
end
for $i = 1 : m$
    for $j = 1 : n$
        if ($i = j$ && $R(i, j) \sim= R(j, i)$)
            Sym = 0;
        end
    end
end

```
for i = 1 : m
    for j = 1 : n
        for k = 1 : n
            if ((R(i,k) == 1) && (R(k,j) == 1) && (R(i,j) == 0))
                Testtrans = 0;
            end
        end
    end
 end
if (ref == 1) && (Sym == 1) && (Testtrans == 1)
    fprintf('R est une relation d''équivalence \n')
else
    fprintf('R n''est  pas une relation d''équivalence car : \n')
if ref == 0
    fprintf('La relation R n''est pas reflexive \n')
end
if Sym==0
    fprintf('La relation R n''est pas symétrique \n')
end
if Testtrans==0
    fprintf('La relation R n''est pas transitive \n')
end
end
```

Exemple 2.12



2.4.2 Relation d'ordre

L'algorithme suivant vérifie que si une relation donnée est une relation d'ordre ou non.

function[ ] = Ordre(R)

clc;

ref = 0; Ansym = 1; Testtrans = 1; $[m, n]$ = size(R);

if (diag(R) == ones(n, 1))

    ref = 1;

end

for $i = 1 :\ m$

    for $j = 1 :\ n$

        if ($i\ = j$ && $R(i, j) == 1$ && $R(j, i) == 1$)

            Ansym = 0;

        end

    end

end

for $i = 1 :\ m$

    for $j = 1 :\ n$

        for $k = 1 : n$

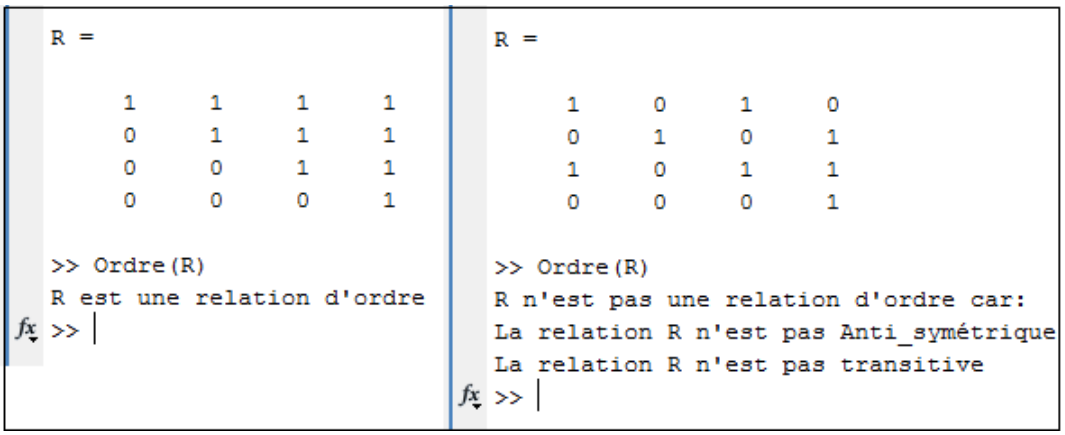
            if (($R(i, k) == 1$) && ($R(k, j) == 1$) && ($R(i, j) == 0$))

                Testtrans = 0;

            end

```
        end
    end
end
if (ref == 1) && (Ansym == 1) && (Testtrans == 1)
    fprintf('R est une relation d''ordre \n')
else
    fprintf('R n''est  pas une relation d''ordre car : \n')
if ref == 0
    fprintf('La relation R n''est pas reflexive \n')
end
if Ansym == 0
    fprintf('La relation R n''est pas Anti-symétriqu \n')
end
if Testtrans == 0
    fprintf('La relation R n''est pas transitive \n')
end
end
end
```

Exemple 2.13

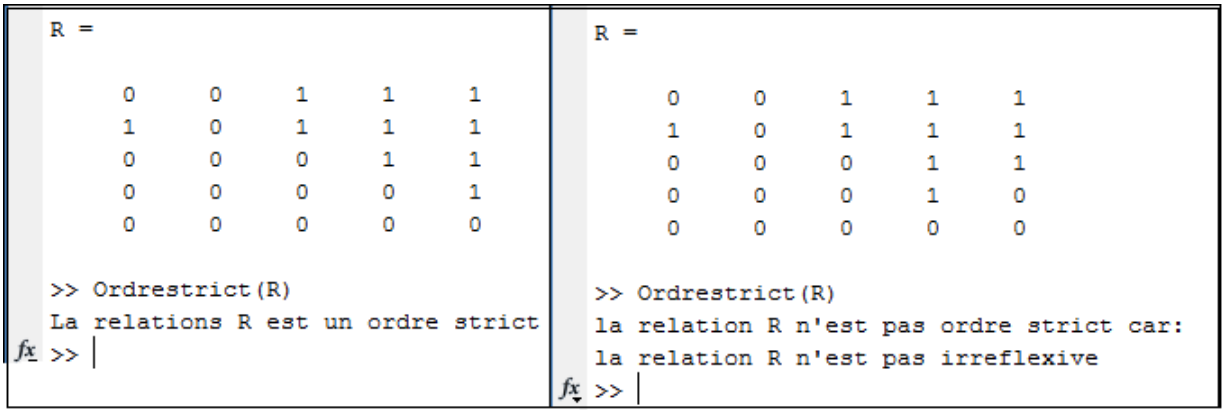


2.4.3 Ordre strict

L'algorithme suivant vérifie que si une relation donnée est une relation d'ordre strict ou non.

```
function[ ] = Ordrestrict(R)
clc; [m, n] = size(R); Testtrans = 1; irref = 0;
if diag(R) == zeros(n, 1)
    irref = 1;
end
for i = 1 : n
    for j = 1 : n
        for k = 1 : n
            if((R(i, k) == 1)&&(R(k, j) == 1)&&(R(i, j) == 0))
                Testtrans = 0;
            end
        end
    end
end
if (irref == 1)&&(Testtrans == 1)
    fprintf('La relations R est un ordre strict\n')
else
    fprintf('La relations R n"est pas ordre strict car :\n')
    if diag(R) ~= zeros(n, 1)
        fprintf('La relations R n"est pas irreflexive\n')
    end
    if Testtrans ~= 1
        fprintf('La relations R n"est pas transitive\n')
    end
end
end
```

Exemple 2.14



2.4.4 Préordre

L'algorithme suivant vérifie que si une relation donnée est une relation ou non.

function[ ] = Preordre(R)

clc; $[m, n]$ = size(R); Testtrans = 1; ref = 0; sym = 0; ansym = 0;

if diag(R) == ones(n, 1)

ref = 1;

end

for $i = 1 : n$

for $j = 1 : n$

if ($R(i, j) \sim= R(j, i)$)

sym = 1;

end

if (($i \sim= j$)&&($R(i, j) == 1$)&&($R(j, i) == 1$))

ansym = 1;

end

end

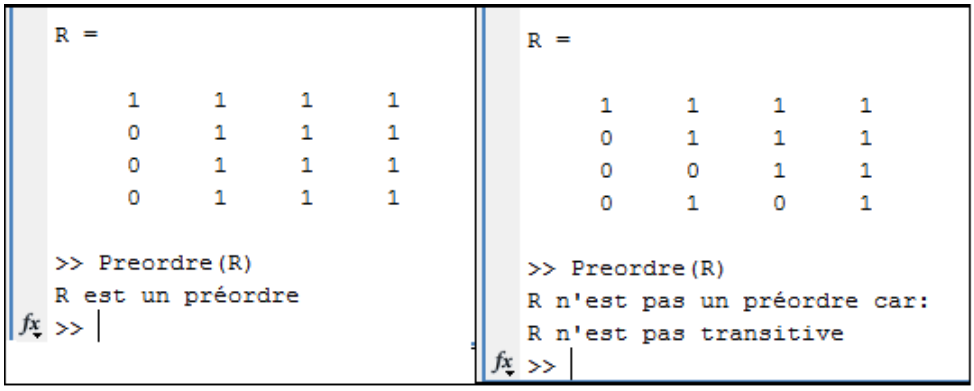
end

for $i = 1 :\ n$

for $j = 1 :\ n$

```
      for k = 1 : n
       if ((R(i,k) == 1) && (R(k,j) == 1) && (R(i,j) == 0))
          Testtrans = 0;
       end
      end
    end
end
 if (ref == 1&&sym == 1&&ansym == 1&&Testtrans == 1)
      fprintf('R est un Préordre \n')
 else
      fprintf('R n''est pas un Préordre car :\n')
          if ref = 0
              fprintf('R n''est pas réflexive\n')
          end
          if sym = 0
              fprintf('R est symétrique\n')
          end
          if ansym = 0
              fprintf('R est anti-symétrique\n')
          end
    if Testtrans == 0
     fprintf('R n''est pas transitive \ n')
    end
end
end
```

Exemple 2.15



2.5 Nombres des relations sur des ensembles finis

Dans cette section, on va donner un algorithme avec un exemple d'exécution sur les nombres des relations binaires sur un ensemble finis.

```
function[ ] = BinaryRelatins(E)
N = length(E); R = zeros(2^(N^2), N^2);  [n, m] = size(R);
v = 0;  res = 0;
for j = 1 :  m
    v = n /(2^j);
      for i = 1 :  n
          res=floor((i − 1) / v);
            if mod(res, 2) == 0
                R(i, j) = 0;
            else
                R(i, j) = 1;
            end
      end
end
%.....................................................................
i = 1; nombre = ' '; verg = ' ';
while i <= length(E)
    if i == 1
        verg = ' ';
```

```
    else
        verg = ' , ';
    end
        nombre = strcat(nombre , verg , E{i});
end
          Ensvar = strcat('lensemble E = {',nombre,'}')
%.....................................................
vect = linspace(1 , 1 , N^2);
str = num2cell(vect);
i = 1;
while i <= N^2
    for k = 1 : N
        for j = 1 : N
            str{i} = strcat('(' , E(k) , ' , ' , E(j) , ')');
                i = i + 1;
        end
    end
end
  str{ : };
 while i <= (length(E))^2
    if i == 1
        verg = ' ';
    else
        verg = ' , ';
    end
        nombre = strcat(nombre , verg , str{i});
            i = i + 1;
 end
  Enspro = strcat('E * E = {',nombre,'}')
%.....................................................
A = zeros(2^(N^2),1);
```

```
B = num2cell(A);
for i = 1 : 2^(N^2)
    Relation = ' ';
      som = ' ';
        s = 1;
    for j = 1 : N^2
        if R(i,j) == 1
        if s == 1
          som=' ';
        else
          som =' , ';
        end
        Relation = strcat( Relation , som , str{j} );
            s = s + 1;
        end
    end
  B{i} = strcat('{ ' , Relation , ' }');
end
    fprintf('les relations possibles sur un ensemble finis de cardinal % d :\n',length(E))
 B{ : }
%.....
mR = zeros(N);
NR = 0;    NIR = 0;    NNR = 0;
%.....
NS = 0;    NAs = 0;    NAns = 0;    NNS = 0;
%.....
Nt = 0;    Nant = 0;    Nnt = 0;
%.....
NC = 0;    NNC = 0;
%.....
ordre = 0;    equiv = 0;
```

```
for z = 1 : n
    Testsym = 1;   Testasym = 1;        Testantisym = 1;
%.....
    Testtrans = 1;   Testantitrans = 1;
%.....
    ref = 0;
%.....
    Testcirc = 1;
        k = 1;          .....% affiche tous les matrices assosie et sont propriété
for l = 1 : N
    for s = 1 : N
        mR(l, s) = R(z, k);
            k = k + 1;
    end
end
 fprintf('\n La relation %  d est :\n',z)
  B{ z }
    fprintf('\n' La matrice associé d"un relation % 2i  est : \n', z)
mR
% .....Reflexivite.....
if (diag(mR) == zeros(N, 1))
    fprintf('La relation % 2i  est irreflexive \n', z)
        NIR=NIR + 1;
elseif (diag(mR) == ones(N, 1))
    fprintf('La relation % 2i  est reflexive \n', z)
        NR=NR + 1;
        ref = 1;
else
    fprintf('La relation % 2i  est non-reflexive \n', z)
        NNR=NNR+1;
end
```

```
% .....Symetrie.....
for i = 1 : N
    for j = 1 : N
        if mR(i, j) ~= mR(j, i)
            testsym = 0;
        end
if ((R(i, j) == 1) && (R(j, i) == 1))
    Testasym = 0;
end
if ((i ~= j) && (R(i, j) == 1) && (R(j, i) == 1))
    Testantisym = 0;
end
    end
 end
    if Testsym == 1
        fprintf('La relation % 2i est symetrique \n', z)
        NS=NS + 1;
    end
  if Testasym == 1
      fprintf('La relation % 2i est asymetrique \n', z)
      NAs=NAs + 1;
  end
    if Testantisym == 1
      fprintf('La relation % 2i est anti-symetrique \n', z)
          NAns=NAns + 1;
    end
      if (Testsym == 0) && (Testasym == 0) && (Testantisym == 0)
      fprintf('La relation % 2i est non symétrique et non anti-symetrique
       non asymetrique au mème temp \n', z)
          NNS=NNS + 1;
      end
```

```
% .....Transitivite.....
for i = 1 : N
    for j = 1 : N
        for k = 1 : N
        if ((mR(i,k) == 1) && (mR(k,j) == 1) && (mR(i,j) == 0))
            Testtrans = 0;
        end
        if ((mR(i,k) == 1) && (mR(k,j) == 1) && (mR(i,j) == 1))
            Testantitrans = 0;
        end
        end
    end
end
if Testtrans == 1
        fprintf('La relation % d est transitive \n', z)
            Nt=Nt+1;
end
if Testantitrans == 1
        fprintf('La relation % d est antitransitive \n', z)
            Nant = Nant+1;
end
if (Testtrans == 0) && (Testantitrans == 0)
        fprintf('La relation % d est non transitive et non antitransitive \n', z)
            Nnt = Nnt+1;
end
% .....circulaire.....
    for j = 1 : N
        for k = 1 : N
            if ((mR(i,k) == 1) && (mR(k,j) == 1) && (mR(j,i) == 0))
                Testcirc = 0;
            end
```

```
        end
    end
end
if Testcirc == 1
    fprintf('La relation % d est circulaire \n', z)
        NC=NC+1;
else
    fprintf('La relation % d n"est pas circulaire \n', z)
        NNC=NNC+1;
end
%.....
if (ref == 1) && (Testsym == 1) && (Testtrans == 1)
    fprintf('La relation % d est une relation d"équivalence \n', z)
        equiv=equiv+1;
end
if (ref == 1) && (Testantisym == 1) && (Testtrans == 1)
    fprintf('La relation % d est une relation d"ordre \n', z)
         ordre=ordre+1;
end
% .....
end
% .....
    fprintf('le nombre des relations binaire de dans un ensemble de % 2i
    elements est % d \n', N, n)
% .....
fprintf('Le nombre des relations reflexive est % 2i \n', NR)
fprintf('Le nombre des relations irreflexive est % 2i \n', NIR)
fprintf('Le nombre des relations non reflexive est % 2i \n', NNR)
% .....
fprintf('Le nombre des relations symetrique est % 2i \n', NS)
fprintf('Le nombre des relations non symetrique et non anti-symetrique et
```

non asymetrique au mtemp est% 2i \n′, NNS)

fprintf(′Le nombre des relations Asymetrique est % 2i \n′, NAs)

fprintf(′Le nombre des relations Anti-symetrique est % 2i \n′, $NAns$)

% .....

fprintf(′Le nombre des relations transitive est % 2i \n′, Nt)

fprintf(′Le nombre des relations Anti-transitive est % 2i \n′, $Nant$)

fprintf(′Le nombre des relations non transitive et non Anti-transitive au m temp est % 2i \n′ , Nnt)

% .....

fprintf(′\ n Le nombre des relations est circulaire est %2i \n, NC)

fprintf(′\n Le nombre des relations n″est pas circulaire est %2i \n′, NNC)

% .....

fprintf(′Le nombre des relations d″équivalence % 2i \n′, $equiv$)

fprintf(′Le nombre des relations d″ordre % 2i \n′, $ordre$)

Exemple 2.16

```
>> E={'x1','x2'}

E =

    'x1'    'x2'

>> BinaryRelatins(E)

Ensvar =

lensemble E={x1,x2}


Enspro =

    'E*E={(x1,x1),(x1,x2),(x2,x1),(x2,x2)}'

les relations possibles sur un ensemble finis de cardinal 2:

ans =

{}


ans =

    '{(x2,x2)}'


ans =

    '{(x2,x1)}'

ans =

    '{(x2,x1),(x2,x2)}'


ans =

    '{(x1,x2)}'


ans =

    '{(x1,x2),(x2,x2)}'


ans =

    '{(x1,x2),(x2,x1)}'


ans =

    '{(x1,x2),(x2,x1),(x2,x2)}'


ans =

    '{(x1,x1)}'


ans =
```

```
    '{(x1,x1),(x2,x2)}'


ans =

    '{(x1,x1),(x2,x1)}'


ans =

    '{(x1,x1),(x2,x1),(x2,x2)}'


ans =

    '{(x1,x1),(x1,x2)}'


ans =

    '{(x1,x1),(x1,x2),(x2,x2)}'


ans =

    '{(x1,x1),(x1,x2),(x2,x1)}'


ans =

    '{(x1,x1),(x1,x2),(x2,x1),(x2,x2)}'
```

```
 La relation   1 est:

ans =

{}


 La matrice associé d'un relation  1 est:

mR =

     0     0
     0     0


ans =

La relation  1 est irreflexive


ans =

La relation  1 est symetrique


ans =

La relation  1 est Asymetrique
```

```
ans =

La relation  1 est Anti_symetrique


La relation 1 est transitive
La relation 1 est Antitransitive
La relation 1 est circulaire

 La relation   2 est:

ans =

    '{(x2,x2)}'


 La matrice associé d'un relation  2 est:

mR =

     0     0
     0     1


ans =

La relation  2 est non reflexive


ans =
```

```
ans =

La relation  2 est symetrique


ans =

La relation  2 est Anti_symetrique


La relation 2 est transitive
La relation 2 est circulaire

 La relation   3 est:

ans =

    '{(x2,x1)}'


 La matrice associé d'un relation  3 est:

mR =

     0     0
     1     0


ans =

La relation  3 est irreflexive
```

```
ans =

La relation  3 est Asymetrique



ans =

La relation  3 est Anti_symetrique


La relation 3 est transitive
La relation 3 est Antitransitive
La relation 3 est circulaire

 La relation   4 est:

ans =

    '{(x2,x1),(x2,x2)}'


 La matrice associé d'un relation  4 est:

mR =

     0     0
     1     1


ans =
```

```
ans =

La relation  4 est non reflexive



ans =

La relation  4 est Anti_symetrique


La relation 4 est transitive
La relation 4 n'est pas circulaire

 La relation   5 est:

ans =

    '{(x1,x2)}'


 La matrice associé d'un relation  5 est:

mR =

     0     1
     0     0


ans =

La relation  5 est irreflexive
```

```
ans =

La relation  5 est Asymetrique



ans =

La relation  5 est Anti_symetrique


La relation 5 est transitive
La relation 5 est Antitransitive
La relation 5 est circulaire

 La relation   6 est:

ans =

    '{(x1,x2),(x2,x2)}'


 La matrice associé d'un relation  6 est:

mR =

     0     1
     0     1


ans =
```

```
ans =

La relation  6 est non reflexive



ans =

La relation  6 est Anti_symetrique


La relation 6 est transitive
La relation 6 n'est pas circulaire

 La relation   7 est:

ans =

    '{(x1,x2),(x2,x1)}'


 La matrice associé d'un relation  7 est:

mR =

     0     1
     1     0


ans =

La relation  7 est irreflexive
```

```
ans =

La relation  7 est symetrique


La relation 7 est Antitransitive
La relation 7 n'est pas circulaire

 La relation   8 est:

ans =

    '{(x1,x2),(x2,x1),(x2,x2)}'


 La matrice associé d'un relation  8 est:

mR =

     0     1
     1     1


ans =

La relation  8 est non reflexive



ans =

La relation  8 est symetrique
```

```
La relation 8 est non transitive et non Antitransitive en mème temp
La relation 8 n'est pas circulaire

 La relation   9 est:

ans =

    '{(x1,x1)}'


 La matrice associé d'un relation  9 est:

mR =

     1     0
     0     0


ans =

La relation  9 est non reflexive



ans =

La relation  9 est symetrique



ans =
```

```
La relation  9 est Anti_symetrique


La relation 9 est transitive
La relation 9 est circulaire

 La relation  10 est:

ans =

    '{(x1,x1),(x2,x2)}'


 La matrice associé d'un relation 10 est:

mR =

     1     0
     0     1


ans =

La relation 10 est reflexive



ans =

La relation 10 est symetrique
```

```
ans =

La relation 10 est Anti_symetrique


La relation 10 est transitive
La relation 10 est circulaire

 La relation 10 est une relation d'équivalence

 La relation 10 est une relation d'ordre

 La relation  11 est:

ans =

    '{(x1,x1),(x2,x1)}'


 La matrice associé d'un relation 11 est:

mR =

     1     0
     1     0


ans =

La relation 11 est non reflexive
```

```
ans =

La relation 11 est Anti_symetrique


La relation 11 est transitive
La relation 11 n'est pas circulaire

 La relation  12 est:

ans =

    '{(x1,x1),(x2,x1),(x2,x2)}'


 La matrice associé d'un relation 12 est:

mR =

     1     0
     1     1


ans =

La relation 12 est reflexive



ans =

La relation 12 est Anti_symetrique

La relation 12 est transitive
La relation 12 n'est pas circulaire

 La relation 12 est une relation d'ordre

 La relation  13 est:

ans =

    '{(x1,x1),(x1,x2)}'


 La matrice associé d'un relation 13 est:

mR =

     1     1
     0     0


ans =

La relation 13 est non reflexive



ans =

La relation 13 est Anti_symetrique
```

```
La relation 13 est transitive
La relation 13 n'est pas circulaire

 La relation  14 est:

ans =

    '{(x1,x1),(x1,x2),(x2,x2)}'


 La matrice associé d'un relation 14 est:

mR =

     1     1
     0     1


ans =

La relation 14 est reflexive



ans =

La relation 14 est Anti_symetrique


La relation 14 est transitive
La relation 14 n'est pas circulaire

 La relation 14 est une relation d'ordre

 La relation  15 est:

ans =

    '{(x1,x1),(x1,x2),(x2,x1)}'


 La matrice associé d'un relation 15 est:

mR =

     1     1
     1     0


ans =

La relation 15 est non reflexive



ans =

La relation 15 est symetrique


La relation 15 est non transitive et non Antitransitive en même temp
La relation 15 n'est pas circulaire

 La relation  16 est:
```

```
 La relation  16 est:

ans =

    '{(x1,x1),(x1,x2),(x2,x1),(x2,x2)}'


 La matrice associé d'un relation 16 est:

mR =

     1     1
     1     1


ans =

La relation 16 est reflexive


ans =

La relation 16 est symetrique


La relation 16 est transitive
La relation 16 est circulaire

 La relation 16 est une relation d'équivalence
```

```
le nombre des relations binaire  dans un ensemble de 2 elements est    16

Le nombre des relations reflexive est  4

Le nombre des relations irreflexive est  4

Le nombre des relations non reflexive est  8

Le nombre des relations symetrique est  8

Le nombre des relations non symetrique est  0

Le nombre des relations Asymetrique est  3

Le nombre des relations Anti_symetrique est 12

Le nombre des relations transitive est 13

Le nombre des relations Anti_transitive est  4

Le nombre des relations non transitive et non Antitransitive en même temp est  2

Le nombre des relations n'est pas circulaire est  7

Le nombre des relations n'est pas circulaire est  9

Le nombre des relations d'équivalenc est  2

Le nombre des relations d'ordre est  3

fx >>
```

Bibliographie

[1] C. Antonini, J. F. Quint, P. Borgna, J. Béard, E. Lebeau, E. Souche, A. Chateau, O. Teytaud, *Les Mathématiques pour l'Agrégation*, 2002.

[2] A. Balhadj, *Génération de Treillis et propriétés algébriques*, Mémoire de Magistère, Université de Mouloud Mammeri, Tizi-Ouzou, 2011.

[3] M. Bares, *Pratique du calcul relationnel*, Edilivre, 2016.

[4] N. Caspard, B. Leclerc, B. Monjardet, *Ensembles ordonnés finis : concepts, résultats et usages*, Springer, 2007.

[5] R. Fraissé, *Théorie of relations*, Elsever science publishers, 1986.

[6] L. Frécon, *Éléments de mathématiques discrètes*, Presses Polytechniques et universitaires romandes, 2002.

[7] D. Ker Andrew, *Discrete Mathematics*, Oxford University Computing Laboratory, 2010.

[8] A. Labourel, *Relations*, Université de Provence, http ://www.univ-amu.fr/, 2011.

[9] G. Legendre, *Cours de Mathématiques*, Université de Versailles Saint-Quentin-En-Yvelines, 2003-2004.

[10] S. Lipschutz, *Discrete Mathematics*, Third edition, Mcgra-Whill, 2007.

[11] M. Marchand, *Outils de mathématiques pour l'informaticien*, 2ème édition, DBS Sciences.

[12] J. Mikram, *Module Mathématiques I : Algèbre*, Université Mohammed V, http ://www.fsr.ac.ma/ANO/, 2005-2006.

[13] M. Pouzet, *Théorie de l'ordre : une introduction*, 2004.

[14] S. Roman, *Lattices and Ordred Sets*, Springer, 2008.

[15] W. Shoafe, *Applied discrete Mathematics*, Department of computer sciences and cyber security college of engineering Florida tech, 2014.

[16] http ://www.larousse.fr/dictionnaires/francais, *Dictionnaire de français Larousse*, 07 Mai 2017.

الملخص:

الهدف من هذه المذكرة هو القيام بدراسة حول العلاقات وخصوصا العلاقات الثنائية. حيث قمنا بالتطرق لجانبها النظري وخواصها الأساسية، ثم بعد ذلك قمنا في الفصل الثاني من المذكرة بإنجاز خوارزميات لحساب العمليات على العلاقات الثنائية ودراسة خواصها وحساب عددها وقد قمنا بتطبيقها بواسطة برنامج **Matlab**.

Résumé

Ce mémoire est porté sur les relations, particulièrement, les relations binaires. Nous nous présentons leurs notions de base et leurs propriétés fondamentales.

Dans le deuxième chapitre, on fait des algorithmes pour calculer les opérations sur les relations binaires et de vérification de leurs propriétés. Ces algorithmes sont établir par le Matlab.