Final Project

BUSINESS MANAGEMENT DEGREE

Faculty of Economics and Business,
Universitat de Barcelona

Theory of decision applied to sports management

Author: Gutiérrez Galopa, Alejandro

Director: Gil Lafuente, Ana María
Department: Business Department

Barcelona, June 2017

"Yo pienso que convertir los sentimientos en matemáticas es realmente algo muy complicado y muy hermoso. La tarea del arte es esa, es transformar todo lo que nos ocurre continuamente en símbolos, transformarlo en música, transformarlo en algo que pueda perdurar en la memoria de los hombres.
Nuestro deber es ese, tenemos que cumplir con él, si no nos sentimos muy desdichados."

- Borges

Resumen

Este proyecto tiene como finalidad ser una introducción a la teoría de decisión y, de manera más tangencial, a la matemática de la incertidumbre.

La toma de decisiones no solamente está presente en cualquier situación cotidiana, sino que también tiene una importancia capital en el ámbito empresarial. En muchas ocasiones, estas decisiones tienen un alto componente de incertidumbre inherente a las consecuencias que puede conllevar cada una de las posibles elecciones. La teoría de la decisión, fundamentada en la matemática de la incertidumbre, permite gestionar y analizar estas situaciones mediante algoritmos que pueden facilitar la elección de la alternativa correcta.

El trabajo se estructura en seis capítulos. El primer capítulo es introductorio; se definen los conceptos básicos y se destacan los rasgos diferenciales tanto de la teoría de la decisión como de la matemática de la incertidumbre. Los siguientes cuatro capítulos, del segundo al quinto, están dedicados a los cuatro elementos básicos de la teoría de la decisión; relación, asignación, agrupación y ordenación respectivamente. Por último, en el sexto capítulo se presenta una aplicación de la teoría de la decisión a la gestión deportiva. Primero, se realiza un estudio de inversión en una franquicia de la liga de baloncesto americana (NBA), seguido de una planificación de grupos óptimos de entrenamiento según las características de los jugadores, y finalmente se estudia la repercusión que pueden tener determinadas acciones de marketing.

Palabras clave

Decisión	Incertidumbre
Relación	Asignación
Agrupación	Ordenación

Abstract

The complexity of human interactions and their unpredictability result in an environment of increasing uncertainty. The aim of this project is to provide an overview of the theory of decision by discussing in depth the elements involved and highlighting both the differentiating characteristics and the theory's adaptability to business decisions.
To demonstrate the functionality of fuzzy mathematics, a business case relating to sports management in the context of basketball is studied and resolved using the theory of decision.

Key words

Decision	Uncertainty
Relation	Assignation
Grouping	Order

Acknowledgements

I would like to thank Ana María Gil Lafuente for all her support and tuition during these months, and all those close to me, especially my family, friends and classmates, all of whom have encouraged me and given me inspiration when I needed it most.
My thanks also to Rosemary who has kept my English on the straight and narrow.

Contents

Introduction

Fuzzy mathematics emerged 50 years ago as a new branch within mathematics, and the theories developed around it bring flexibility and adaptation to classical mathematics. Having studied for a double degree in Mathematics and Business Management, my mathematical background had been based on rigorousness and exactitude so I was intrigued by how fuzzy mathematics deals with what initially seemed to be classical mathematical problems. Therefore, I am of the opinion that the topic I have chosen provides the perfect link between my two fields of study and suggests an innovative treatment of Mathematics in the field of decision-making.

Not only in a business context, but also in day-to-day situations, decision is a key word. Focusing on the business world, decision-making has obvious economic consequences and can have important long-term effects that have to be carefully considered before deciding between one option or another. Therefore, making a decision requires a process of analyzing the pros and cons of all the possible choices in a range; and it is precisely this prediction exercise from which complexity stems. The theory of decision, employing fuzzy mathematics, can provide some key knowledge for our decision-making process and take the decision-maker one step ahead of the competition by supplying him with the tools to handle situations with a degree of uncertainty.

The project begins with a presentation of fuzzy mathematics and the theory of decision in the first chapter. The following four chapters, Chapters 2 to 5, are dedicated to the four basic elements which comprise the theory of decision; relation, assignation, grouping and order. Finally, the last chapter expounds on the application of the theory of decision in relation to an NBA franchise.

During recent years, NBA franchises have experienced an increasing number of structural changes; some have relocated to new home cities, others have seen ownership changes, and we have even seen the creation of new franchises.
This is the reason for carrying out a real case study into which is the most attractive NBA franchise to invest in, together with a study of all the possible marketing actions to take in order to update the image of the franchise, along with a workout plan for the upcoming NBA Draft. The entire study is carried out using the elements and algorithms from the theory of decision.

Chapter 1

Context

This chapter will set the framework in which mathematics of uncertainty and fuzzy sets arise. Beginning with a comparison between classical mathematics and mathematics of uncertainty, the principle of excluded middle and the principle of gradual simultaneity will be presented. Finally, the four fundamental elements of the theory of decision will be discussed.
The objective of this first chapter is to introduce the subsequent four chapters which will be dedicated to the four elements; relation, assignation, grouping and order.

1.1 Introduction to uncertainty

Mathematics of uncertainty, also known as fuzzy mathematics, arises from attempts to develop a formal construction to deal with uncertainty and the increasing unpredictability of the social environment directly affected by human decisions.
In contrast to the operators relative to classical mathematics, which are known as hard, mathematics of uncertainty incorporates soft operators. While hard operators deal with objective and quantifiable problems, soft operators deal with highly subjective issues and elements which do not necessarily have to be quantifiable.
Due to the capital importance it will have on this project, the MaxMin convolution is presented below:

Definition 1.1. (MaxMin convolution) *Let M_1, M_2 be two fuzzy matrices representing fuzzy relations*[1] *$[\mathcal{R}_1]$ and $[\mathcal{R}_2]$ in $U \times V$ and $V \times W$ respectively.*
The MaxMin convolution of M_1 and M_2 is a fuzzy relation in $U \times W$ such that, for all (u, w) in $U \times W$,

$$MaxMin_{M_1,M_2}(u,w) := Max(Min(M_1(u,v), M_2(v,w))) = \bigvee_v (M_1(u,v) \wedge M_2(v,w)), \quad \forall v \in V$$

The symbol used to represent the MaxMin operator is $\circ$.

This transition from traditional models to uncertain models, i.e., introducing soft operators and uncertain variables, provides a closer fit to the complex realities of nowadays.

[1] The concepts of fuzzy matrix and fuzzy relation will be later discussed in this project.

1.2 Principle of excluded middle and principle of gradual simultaneity

For many years, science has assumed as undeniable the principle of excluded middle.

Proposition 1.2. ***(Principle of excluded middle)*** *A statement can not be true and false at the same time. In other words, a statement is either true or false. Using logical notation,*

$$\neg(p \wedge \neg p)$$

Nowadays, taking into consideration the nature of our environment and the importance of humans in decision-making, principle of excluded middle falls a bit short when it comes to explaining certain phenomena.
This is the gap that the principle of gradual simultaneity tries to fill. Contrary to the principle of excluded middle, it gives the attribute of graduality to proposals.

Proposition 1.3. ***(Principle of gradual simultaneity)*** *A statement can at one and the same time be true and false, on the condition that a degree is assigned to its truth and a degree to its falseness.*

This means that for a certain property we have different degrees of truth (for example, we can scale from 0 to 1, both included) instead of the absolute truth or the absolute falseness (which would correspond to the values 1 and 0 respectively). Observe that, in fact, the principle of excluded middle can be thought of as a particular case of the principle of gradual simultaneity where the only possible grades are the extremes (completely true or completely false). This shows that the principle of gradual simultaneity can cover all the theory behind the principle of excluded middle, and can also encompass situations with a higher degree of complexity. Again, the flexibility of multivalent logic demonstrates that assuming the existence of uncertainty instead of a rigid perspective offers more possibilities when dealing with realistic situations and decision-making scenarios.

1.3 The four elements in the theory of decision

Theory of decision distinguishes four basic elements in decision-making: relation, assignment, grouping and order. It is difficult to imagine any decision-making process where none of these elements is present.
Below are briefly introduced these four elements:

a) ***Relation***

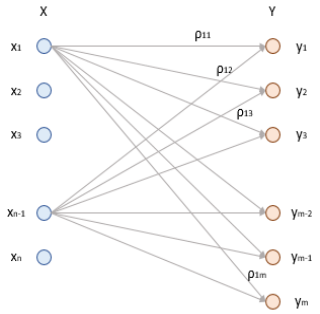
Relation between two agents can be understood as the connection between them. Following the principle of gradual simultaneity, two agents can have different degrees of connection; some of them are stronger and some are weaker. Therefore, although the relation is not a quantifiable element, degrees of relation can be assigned for every two agents of a set.

Remark 1.4. To follow a coherent notation, throughout this paper we will consider a scale from 0 to 1 to measure relation.

Remark 1.5. Note that relation is not necessarily a symmetric property. Considering, for example, a Business Management student and Angelina Jolie, the former will probably know who the latter is, so his degree of relation with the actress can be considered greater than 0, while the latter will probably not know the former, so for her her degree of relation with the student will be 0.

Fuzzy graphs and fuzzy matrices, together with some algorithms, are powerful tools to express and deal with relation between elements or sets.
In this project, special attention will be paid to relations between causes and effects. Given $X = \{x_1, x_2, ..., x_n\}$ the causes and $Y = \{y_1, y_2, ..., y_n\}$ the effects of an event, we can define the relation ρ_{ij} between the cause x_i and the effect y_j.
A typical representation of relations between X and Y by using fuzzy matrices and fuzzy graphs would look like:

$$\begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1m} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & \rho_{nm} \end{pmatrix} \end{array}$$



The process of analyzing the relations between a set of causes $X = \{x_1, x_2, ..., x_n\}$ and a set of effects $Y = \{y_1, y_2, ..., y_n\}$ involves three matrices or graphs:

- matrix M_X, expressing the relations between causes and causes
- matrix $\mathcal{R}_{XY}$, expressing the relations between causes and effects
- matrix M_Y, expressing the relations between effects and effects

Further detail of calculation of relations will be given in Chapter 2, in the forgotten efects matrix explanation.

b) ***Assignation***

In the assignation between two sets, one of them will contain the elements to assign and the other will contain the elements which will receive the assignation. There are three key actors in any assignation, represented by sets:

I) A set representing the elements to assign

II) A set representing the elements which receive the assignment

III) A set representing the assignment criteria

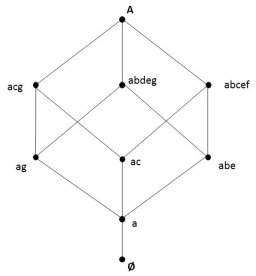
The usual way to proceed with the resolution of assignments is by using fuzzy sets. Further explanation will be given in Chapter 3, where one of the most common algorithms, the Hungarian algorithm, will be detailed step-by-step.

c) ***Grouping***

Normal practice would be to classify the possible options into different groups before making a decision, and this is why grouping is considered a key element in the theory of decision. Basically, the action of grouping revolves around the concept of similarity; two elements can be considered of a same group if they are similar (taking into consideration the pertinent properties). However, similarity is not an easy concept to deal with because of its intransitivity.
This means, defining $\sim_S$ as a similarity relation, and considering a, b and c as three elements of a set,

$$a \sim_S b \ , \ b \sim_S c \quad \not\Longrightarrow \quad a \sim_S c$$

The way to proceed when grouping elements is by building a similarities graph which satisfies the reflexive and symmetric properties. This will reveal the biggest groups of elements with similar characteristics, also known as maximal similarity subrelations.
The graph below shows a typical way of representing grouping for a set of elements $A = \{a, b, c, d, e, f, g\}$, the Galois representation:



In Chapter 4, one of the main methods employed in creating these graphs, the Pichat algorithm, will be studied in-depth. Again, as with relations, connections between elements of two different sets can be measured on a scale (so there are stronger and weaker connections). Using α-cuts, a wide range of boolean matrices can be built and, depending on the criteria assigned, the value of α can change according to the situation. Further explanation will be provided in Chapter 4.

d) ***Order***

Order is the element which culminates the theory of decision. When dealing with unquantifiable variables, traditional modelling tools may fall short; comparing elements and sorting them in a non-quantifiable way can be a productive process. Even when it is not possible to give a strict order for a whole set of elements, equivalence classes can be found (i.e., elements with the same level in the order scale), making it possible to proceed with the ordering of these equivalence classes.
The last section of this chapter will explore the Malgrange algorithm in depth.

Chapter 2

Relation

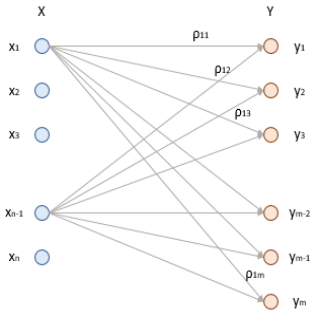
This chapter starts with an introduction of the main concepts relating to relations, including a comparison between fuzzy and boolean relations. Following this, some properties for relations between elements of the same set are defined with the purpose of facilitating the discussion of the connection between relations and grouping and order. The chapter ends with an explanation of a method to compute the forgotten effects in a relation between two sets.

2.1 Basic concepts in relations

Consider $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ two sets of elements and, for each $i \in \{1, ..., n\}$, $j \in \{1, ..., m\}$, consider $\rho_{ij} \in [0, 1]$ the value that measures the relation between x_i and y_j, where 1 represents the higher grade of relation (and 0 the lower one).

The most common way to represent the relations between elements of two sets X and Y is, as was introduced in Chapter 1, by using fuzzy graphs and fuzzy matrices.

$$\begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1m} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & \rho_{nm} \end{pmatrix} \end{array}$$



Fuzzy and boolean visions

From a fuzzy point of view, intensity of a relation between two elements is gradable from 0 to 1. The boolean perspective follows the principle of excluded middle; i.e., the relation exists (and it takes a value of 1) or the relation does not exist (and it takes a value of 0). The following example shows a fuzzy matrix and a boolean matrix.

Example 2.1. Considering $X = \{x_1, x_2, x_3, x_4, x_5\}$ and $Y = \{y_1, y_2, y_3, y_4, y_5\}$ two sets of elements,

$$M_f \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & y_5 \end{array} \\ \begin{pmatrix} 1 & 0.9 & 0.3 & 0 & 0.3 \\ 0.9 & 1 & 0 & 0 & 0.7 \\ 0 & 0.6 & 1 & 0.4 & 0.5 \\ 0 & 0.1 & 0.8 & 1 & 0.6 \\ 0.4 & 0 & 0 & 0.3 & 1 \end{pmatrix} \end{array} \qquad M_B \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array} \begin{array}{c} \begin{array}{ccccc} y_1 & y_2 & y_3 & y_4 & y_5 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

M_f represents a fuzzy relation between the elements in X and Y, while M_B represents a boolean relation between these two sets.

Remark 2.2. A boolean relation can be built from a fuzzy relation. In example 2.1., M_f is a fuzzy relation and M_B is the resulting boolean matrix with 1 representing $\rho_{ij} \geq 0.7$ and 0 representing $\rho_{ij} < 0.7$.
This is a frequent practice, especially with similarity relations. A significance level α is determined, $\alpha \in (0, 1]$, and it is considered that the relation exists when its value ρ_{ij} is higher than or equal to α, and non-existent in the other case. This value α is known as α-cut.
Defining μ_{ij} as the coefficients of the boolean relation (and ρ_{ij} the coefficients of the fuzzy relation) and a significance level α, then, for all i,j,

$$\mu_{ij} = \begin{cases} 0 & , \ \rho_{ij} < \alpha \\ 1 & , \ \rho_{ij} \geq \alpha \end{cases}.$$

2.2 Relation for elements of a same set

Until now two sets X and Y have been used to explain relations between sets. However, there is a particular case where relations between the elements of the same set are studied; i.e., when $X = Y$.
Let $X = \{x_1, x_2, ..., x_n\}$ be a set, and let $\mathcal{R}$ be a relation in $X \times X$. Some properties for relations between elements of a same set are analyzed below:

I) **Reflexitivity**
A relation $[\mathcal{R}]$ is reflexive if

$$\forall i \in \{1, 2, ..., n\} \ , \ \rho_{ii} = 1$$

II) **Symmetry and antisymmetry**
A relation between two elements x_i and x_j does not necessarily have to be symmetric. That means, the intensity of the relation of x_i with x_j can be different to the intensity of the relation of x_j with x_i.
A relation $[\mathcal{R}]$ is symmetric if

$$\forall i, j \in \{1, ..., n\} \ , \ \rho_{ij} = \rho_{ji}$$

A relation $[\mathcal{R}]$ is antisymmetric if

$$\forall i \neq j \in \{1, ..., n\} \ , \ \rho_{ij} \neq \rho_{ji}$$

Therefore, some variations of the "classical" antisymmetry appear in fuzzy relations.
A relation $[\mathcal{R}]$ is a fuzzy antisymmetry if

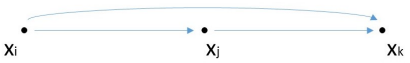
- $\forall \rho_{ij} \in (0, 1], i \neq j \ , \ \rho_{ij} \neq \rho_{ji} \neq 0$,
- $\rho_{ij} = \rho_{ji} = 0$

A relation $[\mathcal{R}]$ is a perfect antisymmetry if

$$\forall i \neq j \ , \ \rho_{ij} > 0 \implies \rho_{ji} = 0$$

Remark 2.3. For boolean relations, fuzzy antisymmetry and perfect antisymmetry are the same concept.

III) **Transitivity**
The intuitive idea of transitivity is that, for three elements $x_i, x_j, x_k \in X$, if x_i and x_j are related, and x_j and x_k are also related, then x_i and x_k are related too.



When dealing with fuzzy relations, the concept of transitivity is slightly more restrictive. Apart from the condition mentioned above, it also has to be satisfied that the direct relation between two elements x_i and x_j has to be greater than or equal to all the indirect relations between them. Formalizing this statement,

$$\forall i, j, k \in \{1, 2, ..., n\} \ , \ \rho_{ik} \geq \bigvee_j (\rho_{ij} \wedge \rho_{jk})$$

Remark 2.4. This means that a fuzzy relation satisfies the transitive property if all the relations between its elements are transitive.

Remark 2.5. Note that to check the transitivity of a fuzzy relation a soft operator (MaxMin convolution) is used.

Remark 2.6. Consider the fuzzy relation $[\mathcal{R}] = (\rho_{ij})$. Its transitivity can be verified when every box of the fuzzy matrix $[\mathcal{R}]^2 := [\mathcal{R}] \circ [\mathcal{R}]$ (the MaxMin convolution of $[\mathcal{R}]$ with itself) is lower than or equal to that in $[\mathcal{R}]$.
In effect, the box ij of $[\mathcal{R}]^2$ (i-th row, j-th column)

$$x_i \begin{pmatrix} \ddots & \dots & \dots & \dots & \dots \\ \vdots & \ddots & \vdots & & \vdots \\ \rho_{i1} & \dots & \rho_{ij} & \dots & \rho_{in} \\ \vdots & & \vdots & \ddots & \vdots \\ \dots & \dots & \dots & \dots & \ddots \end{pmatrix} \circ \begin{pmatrix} & & x_j & & \\ \ddots & \dots & \rho_{1j} & \dots & \dots \\ \vdots & \ddots & \vdots & & \vdots \\ \dots & \dots & \rho_{ij} & \dots & \dots \\ \vdots & & \vdots & \ddots & \vdots \\ \dots & \dots & \rho_{nj} & \dots & \ddots \end{pmatrix}$$

is equal to

$$\bigvee(\rho_{i1} \wedge \rho_{1j}, ..., \rho_{ij} \wedge \rho_{ij}, ..., \rho_{in} \wedge \rho_{nj}) = \bigvee_k (\rho_{ik} \wedge \rho_{kj})$$

Thus, if every box of $[\mathcal{R}]^2$ is lower than or equal to every box of $[\mathcal{R}]$, this means that for all i, j,

$$\rho_{ij} \geq \bigvee_k \rho_{ik} \wedge \rho_{kj}$$

which is the exact definition of transitivity.
Hence $\mathcal{R}$ is a transitive relation if

$$[\mathcal{R}]^2 \leq [\mathcal{R}]$$

Note that, as mentioned previously, boolean relations are a particular case of fuzzy relations, so every definition given about the properties of fuzzy relations can also be applied to boolean relations.

2.3 Relation as a step prior to grouping and order

The four fundamental elements in the theory of decision are not unrelated. When a relation $[\mathcal{R}]$ satisfies at least two of the three properties discussed above (reflexitivity, symmetry and transitivity) grouping and order can be performed.

Definition 2.7. *Consider $[\mathcal{R}]$ a reflexive and symmetric relation.*
Then $[\mathcal{R}]$ is called a ***resemblance relation.***

Definition 2.8. *Consider $[\mathcal{R}]$ a resemblance relation that also satisfies the transitive property (i.e., $[\mathcal{R}]$ is reflexive, symmetric and transitive).*
Then $[\mathcal{R}]$ is called a ***similarity relation.***

Example 2.9.

$$A \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0.6 & 0.1 & 0 \\ 0.6 & 1 & 0.8 & 0.4 \\ 0.1 & 0.8 & 1 & 0 \\ 0 & 0.4 & 0 & 1 \end{pmatrix} \end{array} \qquad B \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0.3 & 0 & 0 \\ 0.3 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0.5 & 1 \end{pmatrix} \end{array}$$

A is a resemblance relation in X. In effect,

- A is reflexive,
$$\forall i \in \{1,2,3,4\}\,, \quad \rho_{ii} = 1$$
- A is symmetric,
$$\forall i,j \in 1,2,3,4\,, \quad \rho_{ij} = \rho_{ji}$$
- A is not transitive,
$$0.1. = \rho_{13} < (\rho_{12} \wedge \rho_{23}) = 0.6 \wedge 0.8 = 0.6$$

B is a similarity relation in X. In effect,

- B is reflexive,
$$\forall i \in \{1,2,3,4\}\,, \quad \rho_{ii} = 1$$
- B is symmetric,
$$\forall i,j \in 1,2,3,4\,, \quad \rho_{ij} = \rho_{ji}$$
- B is transitive,
$$\forall i,j \in 1,2,3,4\,, \quad \rho_{ij} \geq \bigvee_k \rho_{ik} \wedge \rho_{kj}$$

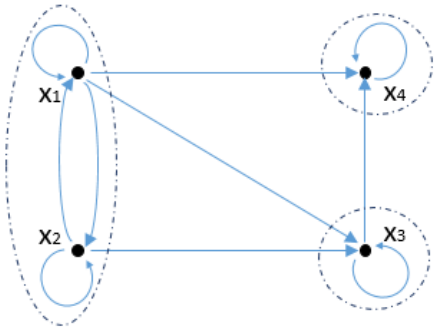
Some definitions in relation to order in relations will now be given.

Definition 2.10. *Consider $[\mathcal{R}]$ a similarity relation without the symmetric property (i.e., $[\mathcal{R}]$ is reflexive and transitive).*
Thus $[\mathcal{R}]$ is called a ***preorder relation.***

The fact that $\mathcal{R}$ is not a symmetric relation opens the possibility of ordering the set.

Example 2.11. Consider the following relation:

$$M \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0.3 & 0.7 & 0.9 \\ 0.1 & 1 & 0.8 & 0.5 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$



The fuzzy graph representation clearly shows the order in this relation,

$$\{x_1, x_2\} \rightarrow \{x_3\} \rightarrow \{x_4\}$$

In a preorder relation similarity subrelations can be found.

Definition 2.12. *Let $\mathcal{R}$ be a preorder relation in a fuzzy set X, and let $Z \subseteq X$ be a subset in X such that, for all $z_i, z_j \in Z$ the property of symmetry is satisfied.*
Thus, the elements in Z form a ***similarity subrelation*** **in** $\mathcal{R}$**.**

Example 2.13. Taking the matrix from the last example above, and giving ρ_{21} the same value as ρ_{12} the new relation clearly preserves the preorder property, and now it has a similarity subrelation in $\{x_1, x_2\}$:

$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} \mathbf{1} & \mathbf{0.3} & 0.7 & 0.9 \\ \mathbf{0.3} & \mathbf{1} & 0.8 & 0.5 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Definition 2.14. *Let $\mathcal{R}$ be a preorder relation which also satisfies the antisymmetry condition.*
Thus, $\mathcal{R}$ is an ***o*****rder relation**.

As seen previously, the antisymmetry property in fuzzy relations is not as clear as it is in boolean relations. The two types of antisymmetry lead to the definition of two different order relations:

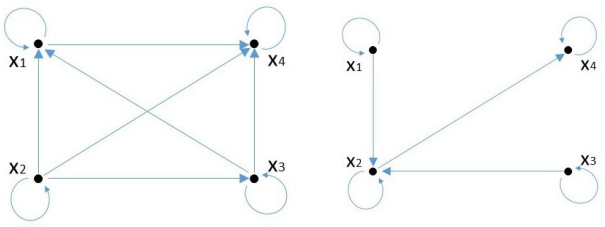
Definition 2.15. *Let $\mathcal{R}$ be a preorder relation which also satisfies the perfect antisymmetry condition.*
Thus, $\mathcal{R}$ is a ***p*****erfect order relation**.

Definition 2.16. *Let $\mathcal{R}$ be a preorder relation which also satisfies the fuzzy antisymmetry condition.*
Thus, $\mathcal{R}$ is a ***n*****on-perfect order relation**.

Example 2.17.

$$M_p \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0.6 \\ 0.6 & 1 & 0.8 & 0.4 \\ 0.1 & 0 & 1 & 0.7 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \qquad M_{np} \equiv \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} 1 & 0.3 & 0 & 0 \\ 0 & 1 & 0 & 0.9 \\ 0 & 0.4 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

M_p is a perfect order relation, with order $\{x_2, x_3, x_1, x_4\}$
M_{np} is a non-perfect order relation where there is no relation, neither direct nor indirect, between x_1 and x_3, so no order between these two elements can be determined. Therefore, the order in this relation is given by two chains; $\{x_1, x_2, x_4\}$ and $\{x_3, x_2, x_4\}$.
Note in the graphs below the order relations in M_p and M_{np}.



Different types of relations between sets have been discussed in this chapter. For further reading on relations and their concatenation, which is beyond the central topic of this project, see *Elements for a theory of decision in uncertainty* ([1] in the bibliography).

2.4 Method. Forgotten effects algorithm

This method is used to find the forgotten effects in a relation. Forgotten effects are those relations between causes and effects which are not explicitly described but do exist. Let $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ be the sets representing the causes and effects of an event, and $\mathcal{R}$ the relation between them. Three matrices are necessary to apply this method:

- A matrix M_X showing the relations between causes and causes.

- A matrix $M_{\mathcal{R}}$ showing the relations between causes and effects.

- A matrix M_Y showing the relations between effects and effects.

Steps in the algorithm

I) Apply the MaxMin convolution between M_X and $M_{\mathcal{R}}$

$$M_{conv} = M_X \circ M_{\mathcal{R}}$$

II) Apply the MaxMin convolution between M_{conv} and M_Y

$$M^*_{\mathcal{R}} = M_{conv} \circ M_Y$$

III) The matrix $M_{\mathcal{R}^*}$ shows the cumulative effects, i.e., both the direct and indirect relations between causes and effects. Therefore, net indirect relations are obtained by substracting the direct relations $M_{\mathcal{R}}$ to $M_{\mathcal{R}^*}$

$$M_F = M^*_{\mathcal{R}} - M_{\mathcal{R}}$$

The values in M_F represent the forgotten effects in the relation $\mathcal{R}$.

Remark 2.18. Take into account that the relation between an agent and itself is always 1.

Chapter 3

Assignation

This chapter begins introducing the notion of assignation and the elements involved in any process of assignation. Also in the first section some important concepts are discussed, such as Hamming distance and weighted assignations.
The purpose of the second section is to present Konig's theorem, which is the basis of the Hungarian algorithm, analyzed in the last section.

3.1 Basic concepts and valuations in assignations

Definition 3.1. *Consider X and Y two sets.* **Assignation** *is the process by which each element in X is ascribed to an element in Y following some specific criteria.*

In an assignation three sets are involved:

1. A set $X = \{x_1, x_2, ..., x_n\}$ containing all the objects to assign
2. A set $Y = \{y_1, y_2, ..., y_m\}$ containing all the objects to receive the assignation
3. A set $C = \{c_1, c_2, ..., c_r\}$ containing all the relevant criteria for the assignation

For every element in X and Y, values (between 0 and 1) for each criteria in C can be defined depending on their characteristics. For example, if only three criteria are considered, x_1 can have a rating of 0.8 for c_1, 0.5 for c_2 , and 0.2 for c_3.
Proceeding in this way for every element in X and Y, sets X and Y can be represented as showing the characteristics of every element:

$$\begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & ... & ... & c_r \end{array} \\ \begin{pmatrix} \rho^X_{n1} & \rho^X_{n2} & \cdots & \cdots & \rho^X_{nr} \\ \rho^X_{n1} & \rho^X_{n2} & \cdots & \cdots & \rho^X_{nr} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho^X_{n1} & \rho^X_{n2} & \cdots & \cdots & \rho^X_{nr} \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & ... & ... & c_r \end{array} \\ \begin{pmatrix} \rho^Y_{n1} & \rho^Y_{n2} & \cdots & \cdots & \rho^Y_{mr} \\ \rho^Y_{n1} & \rho^Y_{n2} & \cdots & \cdots & \rho^Y_{mr} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \rho^Y_{n1} & \rho^Y_{n2} & \cdots & \cdots & \rho^Y_{mr} \end{pmatrix} \end{array}$$

These matrices are the first step towards beginning the assignation between the elements in set X and the elements in set Y.

Remark 3.2. If $n > m$, this means there are more elements in X than in Y. Therefore, some elements in the first set will not have any assigned element in Y. On the contrary, if $n < m$, there will be some elements in Y which will not receive assignation.

To analyze which pairs of elements are the best for the assignation, the notion of distance is used. In this paper the most typical measure, the Hamming distance, will be employed.

Definition 3.3. *Let x_i and y_j be two elements in sets X and Y respectively, and c_k an element in C. Then, the **Hamming distance** between x_i and y_j for the criteria c_k is defined as*

$$d_k(x_i, y_j) = \mid \rho_{ik}^X - \rho_{jk}^Y \mid$$

For all the elements in C, the Hamming distance between x_i and y_j is

$$d(x_i, y_j) = \sum_k \mid \rho_{ik}^X - \rho_{jk}^Y \mid$$

Using the Hamming distance, it is easy to define the relative Hamming distance as

$$\delta_{ij} := \frac{d(x_i, y_j)}{n}$$

Frequently, as the aim of the process is to find the optimal assignation, it is easier to work with coefficients that measure closeness rather than distance. Therefore, the adequacy coefficient f between x_i and y_j for a criteria c_k is defined as

$$f_k(x_i, y_j) := 1 \wedge (1 - \mid \rho_{ik}^X - \rho_{jk}^Y \mid)$$

For all the criteria,

$$f(x_i, y_j) := \sum_k 1 \wedge (1 - \mid \rho_{ik}^X - \rho_{jk}^Y \mid)$$

As for the Hamming distance notion, the relative adequacy coefficient is calculated as

$$\varphi_{ij} = \frac{f(x_i, y_j)}{n}$$

Remark 3.4. The process of assigning ratings to every element for each criteria and then building the matrix with the adequacy coefficients will be done in Chapter 6.

Weighting the criteria in the assignation

Once the adequacy coefficients have been calculated, assignation can be performed. However, using this matrix directly implies that every criteria has the same weight in the assignation. To solve this limitation in the assignation process, some weights can be introduced in order to give more importance to certain criteria than others.
For every element in X, consider the following matrix

$$M_i^X = \begin{array}{c} \\ y_1 \\ y_2 \\ \vdots \\ y_m \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & \dots & c_r \end{array} \\ \begin{pmatrix} f_1(x_i, y_1) & f_2(x_i, y_1) & \dots & f_2(x_i, y_1) \\ f_1(x_i, y_2) & f_2(x_i, y_2) & \dots & f_2(x_i, y_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_i, y_m) & f_2(x_i, y_m) & \dots & f_1(x_i, y_m) \end{pmatrix} \end{array}$$

which contains the adequacy coefficients between the element x_i and every element y_j for every criteria c_k.
Then, a second matrix is built which shows the level of importance of every criteria c_k in relation to every element y_j.

$$W = \begin{array}{c} \\ c_1 \\ c_2 \\ \vdots \\ c_r \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \omega_{11} & \omega_{12} & \dots & \omega_{1m} \\ \omega_{21} & \omega_{22} & \dots & \omega_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{1r} & \omega_{2r} & \dots & \omega_{rm} \end{pmatrix} \end{array}$$

Remark 3.5. Values are taken so that

$$\sum_{j=1}^{n} \omega_{jk} = 1$$

Sum-product composition between a matrix M_i^X and W will result in a vector with the new adequacy coefficients between the element x_i and all the elements in Y:

$$M_i^X \tilde{\circ} W = \begin{array}{c} \\ y_1 \\ y_2 \\ \vdots \\ y_m \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & \cdots & c_r \end{array} \\ \begin{pmatrix} f_1(x_i, y_1) & f_2(x_i, y_1) & \dots & f_2(x_i, y_1) \\ f_1(x_i, y_2) & f_2(x_i, y_2) & \dots & f_2(x_i, y_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(x_i, y_m) & f_2(x_i, y_m) & \dots & f_1(x_i, y_m) \end{pmatrix} \end{array} \tilde{\circ} \begin{array}{c} \\ c_1 \\ c_2 \\ \vdots \\ c_r \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \omega_{11} & \omega_{12} & \dots & \omega_{1m} \\ \omega_{21} & \omega_{22} & \dots & \omega_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \omega_{1r} & \omega_{2r} & \dots & \omega_{rm} \end{pmatrix} \end{array} =$$

$$= x_i \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \gamma_{i1} & \gamma_{i2} & \dots & \gamma_{im} \end{pmatrix} \end{array}$$

where γ_{ij} are the new adequacy coefficient.
Finally,putting all the vectors into rows results in a matrix with the new adequacy coefficients:

$$\Gamma = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1m} \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_{n1} & \gamma_{n2} & \cdots & \gamma_{nm} \end{pmatrix} \end{array}$$

3.2 Konig's contribution to the theory of assignation

This section provides an introduction to Konig's theory, and is followed by the explanation of the Hungarian assignation algorithm in the next section. The algorithm was developed by Harold Kuhn[1], but it was based on the works of Dénes Konig and Jeno Egerváry, two Hungarian mathematicians; the algorithm is named Hungarian algorithm

[1]Harold Kuhn (July 29th, 1925 - July 2nd, 2014) was an American mathematician and Professor Emeritus at Princeton University who developed his career around Game Theory

in honor of these two mathematicians. In the following lines some definitions will precede the formulation of Konig's theorem.
Consider two sets $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ and a set of relations r_{ij}, for all $i \in \{1, ..., n\}$ and $j \in \{1, ..., m\}$, represented in matrix form:

$$\tau = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1m} \\ r_{21} & r_{22} & \dots & r_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \dots & r_{nm} \end{pmatrix} \end{array}$$

Definition 3.6. *A* **support** *of τ is defined as a combination of rows and columns which, when eliminated, leaves the matrix with no zeros. The* **minimum support** *is the support of the matrix with the minimum number of rows and columns.*

Definition 3.7. *The* **dissemination index** *of τ is the minimum number of rows and columns of a support. It is denoted by $D(\tau)$.*

Definition 3.8. *Consider $k \in \mathbb{N}$. The* **connection of k rows and k columns** *of τ is the set of k zeros placed on the intersection of the k different rows and columns.*

Definition 3.9. *The* **binding index** *of τ is the connection with the maximum number of zeros. It is denoted by $Q(\tau)$.*

Following on from these definitions, Konig's theorem can be introduced:

Theorem 3.10. (Konig's theorem) *The dissemination index is equal to the binding index,*

$$D(\tau) = Q(\tau)$$

This principle is the main basis of the Hungarian algorithm, which finds an optimal assignation between elements of two sets. The algorithm is analyzed in detail in the following section.

3.3 Method. The Hungarian algorithm

The Hungarian algorithm is applied to a matrix expressing relations between two sets; in this project the matrix chosen will be

$$\Lambda = M_1 - \Gamma = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} y_1 & y_2 & \cdots & y_m \end{array} \\ \begin{pmatrix} 1-\gamma_{11} & 1-\gamma_{12} & \dots & 1-\gamma_{1m} \\ 1-\gamma_{21} & 1-\gamma_{22} & \dots & 1-\gamma_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 1-\gamma_{n1} & 1-\gamma_{n2} & \dots & 1-\gamma_{nm} \end{pmatrix} \end{array}$$

Remark 3.11. Consider $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ the two sets related to the assignation process. If $n < m$, or $m > n$, the matrix will not be square. In this case, to apply the algorithm, it is necessary to add as many rows or columns as required to transform the initial matrix into a square one. These additional rows or columns are called fictitious rows or fictitious columns.

Steps of the algorithm:

I) Consider the matrix Λ.

- If one or more fictitious rows are added, the process starts subtracting the minimum value in each column from every element in that same column. After this, the process is repeated for the rows.
- If one or more fictitious columns are added, the same process is applied but beginning with the rows.
- If the matrix Λ is square, the process can begin with either rows or columns.

II) If there are enough zeros in the matrix to assign every element in X to an element in Y, an optimal assignation has been found. If not, the process continues as follows:

III) The minimum number of rows and columns containing all the zeros is found. These rows and columns are then crossed out.

IV) The lowest value remaining in the matrix is then subtracted from all non crossed out elements, and added to the crossed out elements.

V) With the new matrix created in step IV), the algorithm goes back to step II).

Remark 3.12. Generally, there is no one solution and different assignations may be found.

Chapter 4

Grouping

Chapter 4 first introduces the concept of affinities, as it is a central concept in grouping. Some more important concepts as α-cuts are also introduced in the first section.
Second section analyze Moore closings giving some definitions which facilitate the introduction of the notion of Galois connections, discussed in section three. Finally, the fourth section analyzes two grouping methods; maximum inverse correspondence and Pichat algorithm.

4.1 Basic concepts in grouping

Grouping consists of finding elements in a set which have similar properties or behaviours taking into account some specific attributes. The whole theory of grouping is built on the notion of affinity.

Definition 4.1. ***Affinities*** *are defined as homogeneous agroupations that link elements which are different in nature through their similarities.*

Note that for relations and assignments, fuzzy matrices have been employed. In grouping, however, affinities between elements are expressed in boolean matrices.
Let $X = \{x_1, x_2, ..., x_n\}$ be a set of elements and $C = \{c_1, c_2, ..., c_r\}$ a set of characteristics or properties. The initial matrix representation for analyzing grouping elements is (similarly to when studying assignments)

$$\begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & ... & c_r \end{array} \\ \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1m} \\ v_{21} & v_{22} & \dots & v_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \dots & v_{nm} \end{pmatrix} \end{array}$$

where $v_{ij} \in [0, 1]$ measures the satisfaction of element x_i to property c_j.
Therefore, the process begins with a fuzzy matrix. Once this matrix is built, a threshold or α-cut is set.

α-cuts or thresholds in grouping

Previously in this project the concept of α-cut has been introduced. It consists of fixing a level $\alpha \in [0,1]$ which is considered the acceptance level; this means that all values lower than α are considered zeros, and those greater than or equal to α are ones. For a complete discussion see Remark 2.2. For grouping purposes, different criteria are considered, and applying one α-cut to the matrix presented above means fixing the same acceptance level for every criteria. Depending on the case, it may be useful to fix different α values for each c_j (for example, for c_1 fixing 0.3 as the α-cut, and for c_2 take 0.6 as the significance level). Therefore, instead of fixing a single α for all the criteria, usually r different α_j are fixed, one for every c_r. Thus, applying the α-cuts to the fuzzy matrix results in the following

$$\mathcal{F} = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & ... & c_r \end{array} \\ \begin{pmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1m} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{n1} & \beta_{n2} & \cdots & \beta_{nm} \end{pmatrix} \end{array}$$

where $\beta_{ij} \in \{0,1\}$.

Example 4.2. Consider x_1, x_2, x_3 three candidates for a company for three different positions, and three criteria c_1, c_2, c_3 where c_1 is experience, c_2 is the level of studies and c_3 the social skills.
Consider now the following affiniities matrix

$$\mathcal{B} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \end{array} \begin{array}{c} \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \\ \begin{pmatrix} 0.3 & 0.5 & 0.8 \\ 0.7 & 0.7 & 1 \\ 0.8 & 0.3 & 0.6 \end{pmatrix} \end{array}$$

Adjusting to the particular requirements of the case, a different α-cut is determined for every criteria:

$$\begin{cases} \alpha_1 = 0.6 \\ \alpha_2 = 0.5 \\ \alpha_3 = 0.7 \end{cases}.$$

So, applying these α-cuts to the matrix, the following boolean matrix results:

$$\mathcal{B} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_n \end{array} \begin{array}{c} \begin{array}{ccc} c_1 & c_2 & c_3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \end{array}$$

4.2 Moore closings and their application to fuzzy graphs

Definition 4.3. *Given a set $X = \{x_1, x_2, ..., x_n\}$, its* ***power set*** *is the set containing all the possible combinations of elements*

$$\mathcal{P}(X) = \{\emptyset, x_1, ..., x_n, x_1x_2, ..., x_1x_n, ..., X\}$$

Definition 4.4. *A* **Moore family** $\mathcal{F}(X)$ *is a set of elements such that*

a) $X \subset \mathcal{F}(X)$

b) $Y \in \mathcal{F}(X)$, $Z \in \mathcal{F}(X) \Rightarrow Y \cap Z \in \mathcal{F}(X)$

Definition 4.5. *Let* X *be a set and* $M : X \to X$ *a functional mapping.* M *is a* **Moore closing** *if* M *satisfies the following properties*

a) extensivity

$$\forall Y \in \mathcal{P}(X) \, , \; Y \subset M(Y)$$

b) idempotence

$$\forall Y \in \mathcal{P}(X) \, , \; M(M(Y)) = M(Y)$$

c) isotony

$$\forall Y, Z \in \mathcal{P}(X) \; ; \; Y \subset Z \Rightarrow M(Y) \subset M(Z)$$

Example 4.6. Consider the set $X = \{x_1, x_2, x_3\}$.
A common way to represent a Moore closing M for the power set $\mathcal{P}(X)$ is by using the matrix notation. For example:

$$M = \begin{array}{c} \\ \emptyset \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ X \end{array} \begin{array}{c} \begin{array}{cccccccc} \emptyset & x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 & x_2x_3 & X \end{array} \\ \left(\begin{array}{cccccccc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \right) \end{array}$$

After defining the notion of Moore closing, a basic concept for applying grouping algorithms is introduced below. Let $X = \{x_1, x_2, ..., x_n\}$ be a set, and $\mathcal{B} = (\beta_{ij})$ the boolean matrix of a relation. Then,

Definition 4.7. *It is defined as* **connection to the right** B^+ *the subset of elements* A *in* X *such that*

$$\forall x \in A \, , \; B^+(A) := \{p \in X \;\; \beta_{xp} = 1\} \;\; , \;\; B^+(\emptyset) := X$$

Definition 4.8. *It is defined as* **connection to the left** B^- *the subset of elements* A *in* X *such that*

$$\forall x \in A \, , \; B^-(A) := \{p \in X \;\; \beta_{px} = 1\} \;\; , \;\; B^-(\emptyset) := X$$

Example 4.9. This example will help explain the two concepts defined above. Consider $X = \{x_1, x_2, x_3\}$ and $\mathcal{B} = (\beta_{ij})$ a boolean relation in X.

$$\mathcal{B} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \end{array} \begin{array}{c} \begin{array}{ccc} x_1 & x_2 & x_3 \end{array} \\ \left(\begin{array}{ccc} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{array} \right) \end{array}$$

Then,

$B^+(\emptyset) = X$ $B^+(x_1) = \{x_1, x_3\}$ $B^+(x_2) = X$
$B^+(x_3) = \{x_1\}$ $B^+(x_1, x_2) = \{x_1, x_3\}$ $B^+(x_1, x_3) = \{x_1\}$
$B^+(x_2, x_3) = \{x_1\}$ $B^+(X) = \{x_1\}$
$B^-(\emptyset) = X$ $B^-(x_1) = X$ $B^-(x_2) = \{x_2\}$
$B^-(x_3) = \{x_1, x_2\}$ $B^-(x_1x_2) = \{x_2\}$ $B^-(x_1, x_3) = \{x_1, x_2\}$
$B^-(x_2, x_3) = \{x_2\}$ $B^-(X) = \{x_2\}$

The following proposition provides a way to find Moore closings from connections to the right and to the left:

Proposition 4.10. *Consider R a fuzzy relation in a set X, and B^+, B^- its connections to the right and to the left respectively.*
Then,

$$M_1 = B^+ \circ B^- \ , \ M_2 = B^- \circ B^+$$

are Moore closings of X.

Remark 4.11. The demonstration of the above proposition goes beyond the purposes of this project.

Definition 4.12. *Let X be a set. A* **closed subset** *A in $\mathcal{P}(X)$ is a subset that satisfies*

$$M(A) = A$$

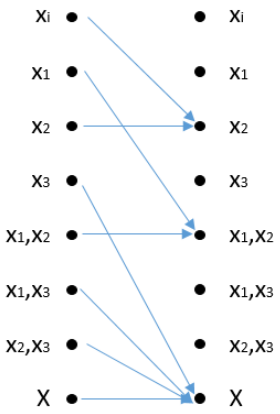
Example 4.13. This example shows the full process required to find the representation of the subsets of closed elements.

$$B^+ \circ B^- = \begin{array}{c} \\ \emptyset \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ X \end{array} \begin{array}{c} \begin{array}{cccccccc} \emptyset & x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 & x_2x_3 & X \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} \circ \begin{array}{c} \\ \emptyset \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ X \end{array} \begin{array}{c} \begin{array}{cccccccc} \emptyset & x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 & x_2x_3 & X \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array} =$$

$$= \begin{array}{c} \\ \emptyset \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ X \end{array} \begin{array}{c} \begin{array}{cccccccc} \emptyset & x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 & x_2x_3 & X \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Thus, a Moore closing for the boolean relation $\mathcal{B}$ is

$M(\emptyset) = x_2$
$M(x_1) = \{x_1, x_2\}$
$M(x_2) = x_2$
$M(x_3) = \{x_1, x_2\}$
$M(x_1, x_2) = x_2$
$M(x_1, x_3) = \{x_1, x_2\}$
$M(x_2, x_3) = x_2$
$M(X) = \{x_1, x_2\}$



Proceeding the same way, $B^- \circ B^+$ can be calculated:

$$B^- \circ B^+ = \begin{array}{c} \\ \emptyset \\ x_1 \\ x_2 \\ x_3 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ X \end{array} \begin{array}{c} \begin{array}{cccccccc} \emptyset & x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 & x_2x_3 & X \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Now, closed subsets for $M_1 = B^+ \circ B^-$ and $M_2 = B^- \circ B^+$ can be found:

$$\bigcup_{A \subset \mathcal{P}(X)} B^+(A) = \{\{x_1\}, \{x_1, x_3\}, X\}$$

$$\bigcup_{A \subset \mathcal{P}(X)} B^-(A) = \{\{x_2\}, \{x_1, x_2\}, X\}$$

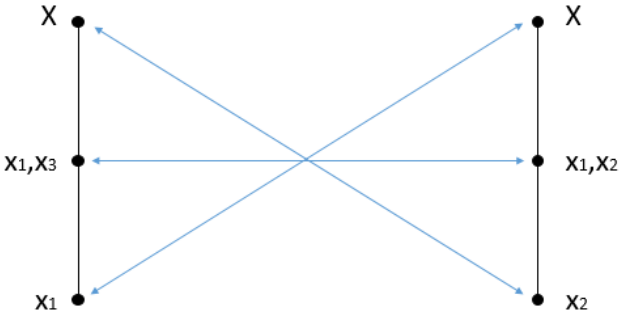
These are the closed subsets for M_1 and M_2 respectively, and can be represented as follows:

Observe that for these two subsets B^+ and B^- are bijective and inverse functions. In effect,

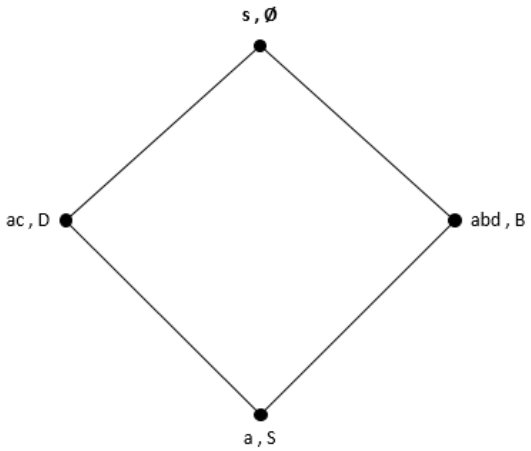
$$B^-(x_1) = X \quad , \quad B^-(\{x_1, x_3\}) = \{x_1, x_2\} \quad , \quad B^-(X) = \{x_2\}$$

$$B^+(x_2) = X \quad , \quad B^+(\{x_1, x_2\}) = \{x_1, x_3\} \quad , \quad B^+(X) = \{x_1\}$$

Graphically:



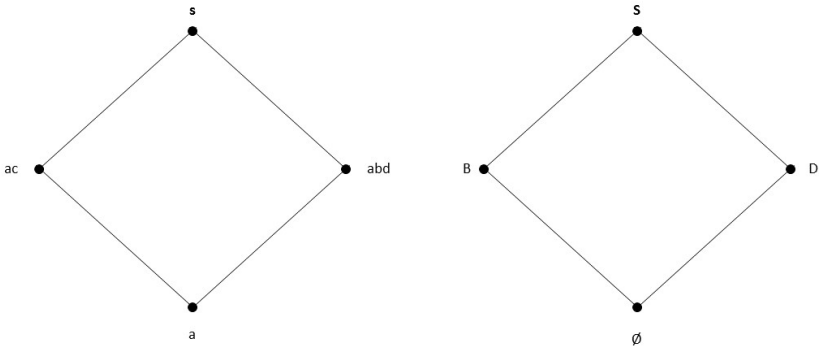
Now, considering the bijective and inverse properties of B^+ and B^-, the representation can be done in a single reticle:



4.3 Galois connections between different sets

This section aims to show the case where the Moore closing M is a mapping from a set X to a set Y. Previously, a Moore closing has been defined as a functional mapping from X to X, but the two sets can be different.
Therefore, if $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ are two different sets and $n \neq m$ matrices B^+ and B^- will not be square matrices, but the process is exactly the same as that analyzed in the previous section when $X = Y$.

Example 4.14. To avoid repeating exactly the same process as that in Example 4.13., assume the representation of the closed subsets found for M_1 and M_2 for a relation between two sets $s = \{a, b, c, d\}$ and $S = \{A, B, C, D\}$ is



Observe that when expressing the affinities, when moving from one group to another and the number of elements in a set increases, the number of elements of the other set decreases, and vice versa.
A ***Galois reticle*** is built by adding the upper limit $(X, \emptyset)$ and the lower limit $(\emptyset, Y)$ to the reticle. As seen in the example, Galois connections show the elements of both groupings linked. Moreover, this representation gives a very clear picture of order in both sets.

4.4 Method. Maximum inverse correspondence and Pichat algorithm

The ***maximum inverse correspondence algorithm*** is a method used to obtain the Galois reticle for a relation between two different sets. Consider $X = \{x_1, x_2, ..., x_n\}$ and $Y = \{y_1, y_2, ..., y_m\}$ two sets, and $\mathcal{B}$ a boolean relation between them.

Steps in the algorithm

I) Choose the set (X or Y) with a lower number of elements.

II) Build the power set of the set chosen in step I).

III) Calculate the connection to the right if X is the chosen set, or the connection to the left if Y is the chosen set.

IV) For every non-empty element in $B^+(x)$, where $x \in \mathcal{P}(X)$, (or $B^-(y)$, where $y \in \mathcal{P}(Y)$ in the case of having calculated the connection to the left) choose its preimage with a higher number of elements.

V) Build the Galois reticle with the relations found in step IV).

Pichat algorithm is a powerful tool for finding maximal subrelations between elements of a same set $X = \{x_1, x_2, ..., x_n\}$.
The algorithm begins with the matrix $\Psi := (\varphi_{ij})$, where φ_{ij} is the adequacy coefficient between x_i and x_j.

Steps in the algorithm

I) Apply the pertinent α-cuts to convert the matrix Ψ into a boolean matrix $\Psi_B = (\psi_{ij})$.

II) For every element in Ψ_B such that $i \leq j$ assign a 0. The result is an upper triangular matrix Ψ_U with zeros in the diagonal.

III) For every row i in the matrix Ψ_U compute the boolean addition $i + (j_1, ..., j_k)$, where $j_1, ..., j_k$ are all the $j > i$ such that $\psi_{ij} = 0$.

IV) Calculate the boolean product of all the boolean additions computed in step III). If a row i^* does not have any element such that $\psi_{i^*j} = 0$ and $j > i*$, then the value 1 is assigned. The results are expressed in minimal terms; that means, $a + a = a$ and $a + ab = a$.

V) The result of this product is a sum of element agroupations. The complementaries of every agroupation are the maximal subrelations which satisfy transitivity property.

Chapter 5

Order

Following the structure of the previous chapters, Chapter 5 starts by defining some basics for establishing order in a set of elements.
The second section in the chapter introduces the different types of order relation that can be found in a set, followed by the explanation of grouping elements as a previous step to ordering.
The last section analyzes an algorithm for ordering elements; the Malgrange algorithm.

5.1 Basic concepts in order

Definition 5.1. *Order is a scale in the preferences between objects based on some properties and attributes.*

Before establishing an order relation in a set of elements, it is important to carefully select the properties that will determine the order.
Two agents take part in an order relation:

I) The set $X = \{x_1, x_2, ..., x_n\}$ containing the elements to order

II) The set $C = \{c_1, c_2, ..., c_m\}$ containing the ordering criteria (or attributes)

For an element x_i, values $\rho_{ij} \in [0, 1]$ measure the level of each attribute c_j.
The following matrix results:

$$\begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} c_1 & c_2 & ... & c_m \end{array} \\ \begin{pmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1m} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & \rho_{nm} \end{pmatrix} \end{array}$$

From this matrix begins the ordering process. For every criteria c_k a square matrix $\mathcal{O}_k = (\beta_{ij})$ is calculated as follows:

$$\beta_{ij} = \begin{cases} 0 & , \ \rho_{ik} < \rho_{jk} \\ 1 & , \ \rho_{ik} \geq \rho_{jk} \end{cases}.$$

These matrices $\mathcal{O}_k$ show the preferences between elements for every criteria c_k. Now, summing these matrices results in a matrix $\mathcal{O}_\Sigma = (\sigma_{ij})$ which contains values in $\{1, 2, ..., n\}$, where σ_{ij} represents the number of times the element x_i is preferred to x_j (elements in the diagonal are not significant, so they are not considered).

$$\mathcal{O}_\Sigma = \sum_k \mathcal{O}_k$$

To keep mathematical coherence, elements will be divided by m to produce a matrix $\mathcal{O} = (\theta_{ij})$ with coefficients between 0 and 1.

$$\mathcal{O} = \begin{array}{c} \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & \dots & x_n \end{array} \\ \begin{pmatrix} X & \theta_{12} & \dots & \theta_{1n} \\ \theta_{21} & X & \dots & \theta_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{n1} & \theta_{n2} & \dots & X \end{pmatrix} \end{array}, \quad \forall i \neq j,\ \theta_{ij} \in [0, 1]$$

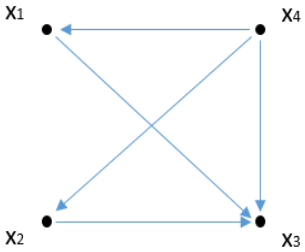
The last step in establishing order between the elements is to apply an α-cut to the matrix $\mathcal{O}$, which determines from which level on it is considered that one element is preferrable to another.
Thus, this can be represented both in a boolean matrix and in a graph.

Example 5.2. Assume that, for a set of elements $X = \{x_1, x_2, x_3, x_4\}$ and for some criteria the matrix $\mathcal{O}$ represents the priorities between elements:

$$\mathcal{O} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} X & 0.5 & 0.75 & 0.5 \\ 0.5 & X & 0.75 & 0 \\ 0.5 & 0.25 & X & 0.25 \\ 0.75 & 1 & 1 & X \end{pmatrix} \end{array}$$

Applying an α-cut equal to 0.6, the representation of boolean preferences is:

$$\mathcal{O} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} X & 0 & 1 & 0 \\ 0 & X & 1 & 0 \\ 0 & 0 & X & 0 \\ 1 & 1 & 1 & X \end{pmatrix} \end{array}$$



5.2 Different types of order relations

The previous section has introduced the first steps for establishing order relations. It has been seen that when the boolean matrix representing priorities between elements contains a 1 in the box (x_i, x_j) this means that x_i is preferred over x_j. However, there is the possibility that the box (x_j, x_i) is also equal to 1. This would mean that x_i is preferred

over x_j, but at the same time x_j is preferred over x_i (in graph theory, this is called a cycle[1] $x_i - x_j$). Thus, order between x_i and x_j cannot be established.
This proves the following simple proposition.

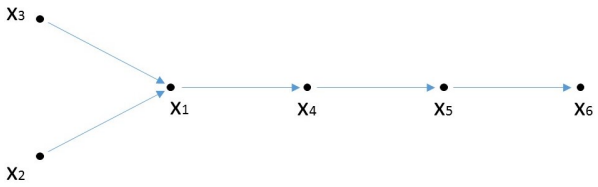
Proposition 5.3. *Let G be the graph of a relation. If there is a cycle in G, then order cannot be established.*

Corollary 5.4. *The existence of cycles in a relation implies the existence of i^*, j^* such that in the matrix $\mathcal{O} = (\theta_{ij})$, $\theta_{i^*j^*} = \theta_{j^*i^*} = 1$.*
Therefore, for the existence of order in relations, antisymmetry is necessary.

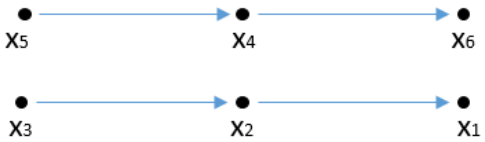
Thus, only antisymmetric matrices will be considered when discussing order relations.
There are three main types of order relations, each with a different nature:

a) Total order relations: linear order can be established for all elements in the set. Graphically:

b) Split or juncture order relations: there is no order relation between two or more elements. Graphically:



c) Disjunct partial order relations: there are two or more disjunct total order relations for subsets in the set. Graphically:



[1] In graph theory, if there is a path leaving from a vertex which comes back to the same vertex, it is known as a cycle.

5.3 Grouping equivalent elements for the ordering

In section 5.2. it has been proved that total order between all the elements of a set cannot be established if there are cycles. In fact, cycles represent equivalence between the elements involved, and therefore it is quite trivial that all the elements in the cycle are at the same level.
Grouping the elements from a cycle is a way of solving this.

Equivalence classes

Definition 5.5. *Let X be a set. An **equivalence class** in X is a subset $A \subseteq X$ such that there is an equivalence relation between any two elements in A; i.e., reflexive, symmetric and transitive properties are satisfied.*

Thus, when finding the equivalence classes in the preferences matrix $\mathcal{O}$ the result is all the groups of elements for which the preference is the same. That means, all the elements of the same equivalence class are equally preferred, so no distinction can be done between them.
If the set has some cycles, first step before ordering is finding the equivalence classes. Once found, instead of ordering the elements of the set, the equivalence classes are ordered. The result of the order process will be the order between equivalence classes, and obviously all the elements which are equally preferred will not be ordered.

Remark 5.6. If a set $X = \{x_1, ..., x_n\}$ has no cycles, when finding the equivalence classes the result will be n different equivalence classes, one of each element. The interpretation of this result is that there are no elements equally preferred, which clearly means that an order can be found without need of grouping elements.

The following section introduces Malgrange algorithm, a method to order the equivalence classes of a set.

5.4 Method. Malgrange algorithm

Malgrange algorithm is a method to find the equivalence classes and its order between the elements of a set.
One of the fastest ways of finding the equivalence classes, and hence their elements, is by employing the procedure explained below.

Steps in the algorithm

Malgrange algorithm is divided in two phases.

Phase I

The first phase aims to find the equivalence classes in a set of elements. Steps followed in this first phase are detailed below, followed by an example to clarify the process:

I) The starting matrix is the boolean matrix $\mathcal{O}$, which represents the priorities between elements. Let $X = \{x_1, x_2, ..., x_n\}$ be the set to order. An arbitrary element x_* is chosen.

II) For x_*, its transitive closing $\Gamma(x_*)$ and its inverse transitive closing $\Gamma^{-1}(x_*)$ are calculated.

III) The intersection $\Gamma(x_*) \cap \Gamma^{-1}(x_*)$ give the equivalence class of x_*.

IV) Rows and columns for all the elements in the equivalence class found in the previous step are eliminated.

V) Now with a smaller matrix, the process is repeated again from step II) until all the rows and columns are eliminated.

Once the process is finished, all the equivalence classes will have been found.

Example 5.7. Consider the set $X = \{x_1, x_2, x_3, x_4\}$, and the boolean matrix below representing the priorities between elements in X:

$$\mathcal{O} = \begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{cccc} x_1 & x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} X & 1 & 1 & 0 \\ 0 & X & 1 & 0 \\ 0 & 1 & X & 0 \\ 1 & 1 & 1 & X \end{pmatrix} \end{array}$$

Equivalence classes are going to be calculated by using the process explained previously:

I) The first element chosen is x_1.

II) Calculating the transitive closings $\Gamma(x_1)$ and $\Gamma^{-1}(x_1)$:

$$\Gamma(x_1) = \{x_1, x_2, x_3\} \quad , \quad \Gamma^{-1}(x_1) = \{x_1, x_4\}$$

III) Then, the intersection $\Gamma(x_1) \cap \Gamma^{-1}(x_1)$ is $\{x_1\}$. Therefore, $\{x_1\}$ is an equivalence class in X.

IV) The row and the column of x_1 are eliminated to continue with the process:

$$\begin{array}{c} \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{ccc} x_2 & x_3 & x_4 \end{array} \\ \begin{pmatrix} X & 1 & 0 \\ 1 & X & 0 \\ 1 & 1 & X \end{pmatrix} \end{array}$$

V) Now the process is repeated for the element x_2. To avoid extending too much the example, results are directly given:

$$\Gamma(x_2) = \{x_2, x_3\} \quad , \quad \Gamma^{-1}(x_2) = \{x_2, x_3, x_4\} \Rightarrow \Gamma(x_2) \cap \Gamma^{-1}(x_2) = \{x_2, x_3\}$$

Then, another equivalence class in X is $\{x_2, x_3\}$.

VI) Finally, $\{x_4\}$ is the other equivalence class in X.

Equivalence classes in X are:

$$\{x_1\}, \{x_2, x_3\}, \{x_4\}$$

This means that x_2 and x_3 are equivalent elements for this relation, and after grouping them an order relation can be established.

Phase II

Once the equivalence classes are found, the second phase of the algorithm aims to order these equivalence classes. The way to proceed to find order does not follow specific steps. Usually, the ordering is found reconstructing the graph by grouping all the elements in the same equivalence class.

Chapter 6

Application to sports management

In this chapter, all the theory of decision seen previously is applied to a sports case. The chapter begins with an introduction to the case, following which the methods discussed in the previous chapters will be applied in order to carry out a study into which is the best NBA franchise to invest in, as well as build a training plan and analyze the effects of various marketing actions.

Background and introduction to the case

National Basketball Association, also known as NBA, is one of the most famous sports leagues in the world. Thirty teams, more commonly known as franchises, participate in this competition. The structure of an NBA franchise is similar to the structure of a big company; the franchise has owners and closes the year with profits or losses. Therefore, operations such as share trading or acquiring franchises are possible, and are becoming more and more common. There are even cases of teams which are listed on the Stock Exchange, such as the Golden State Warriors.
Three recent cases of operations related to NBA franchises are:

a) After the 2003-04 season, the NBA announced the creation of a new franchise; the Charlotte Bobcats. The Bobcats were created by a group of investors, including celebrities such as Larry Bird and the singer Nelly. Their debut in the NBA was the 2004-05 season.

b) Having failed to obtain US$220 million of public funding to remodel their arena, the Seattle Supersonics moved to Oklahoma City in 2008. Once there, the franchise owners decided to change the name of the team by popular vote, and the franchise is now known as Oklahoma City Thunder.

c) Mikhail Prokhorov took an 80% stake in the New Jersey Nets in 2010. Following that, in 2012, he moved the team to Brooklyn, and completed the purchase of the remaining 20% stake in 2015. However, following some poor decisions in relation to the management of his franchise, Prokhorov has now put up for sale 49% of his stake.

Application of the methods

The methods discussed in each chapter of this project will be applied for the following purposes:

- A study of which franchise is the best to invest in will be made by using the Malgrange algorithm.
- Once this franchise has been selected, the Draft workouts will be planned. Further explanation about Draft workouts will be given in the following pages. Pichat and Hungarian algorithm will be the tools employed to plan the workouts.
- Finally, the board of directors wants to promote a strong marketing campaign to boost popularity and update the team image. The forgotten effects algorithm will give some information about which marketing actions could be the most effective.

Selection of the best franchise to buy

A group of investors is considering buying an NBA franchise prior to the start of the 2017-18 season. By using the theory of decision they will try to optimize their decision and analyze which the best team in the NBA is to invest in following some specific criteria. The following pages show the viability analysis they carried out as a first exploration.
The board of directors has chosen 14 criteria on which they will base their selection with respect to the best team to invest in:

c1. Economic situation of the state.

c2. Importance of basketball in the city.

c3. Weather and social conditions.

c4. Tax regulations.

c5. Current situation of the team.

c6. Team projection.

c7. Playoff chances.

c8. Marketing income.

c9. Media income.

c10. Management board stability.

c11. Arena attendance.

c12. Arena facilities.

c13. Popularity in the USA.

c14. Popularity outside the USA.

Remark 6.1. For the purposes of this project, all the criteria have the same weight in the process. However, the process can be as complex as desired, adding weights depending on the importance of each criteria. The appendix contains an explanation of how the valuation for each criteria has been made.

Putting together all the valuations in a matrix:

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}
Celtics	1	0.9	0.7	0.8	0.9	0.9	0.9	0.8	0.9	0.6	0.7	1	0.9	0.9
Nets	1	0.3	0.7	0.8	0	0.2	0	0.6	0.4	0.2	0.1	0.5	0.1	0.2
Knicks	0.9	0.8	0.8	0.8	0.4	0.6	0.5	1	0.6	0.3	0.9	1	0.7	0.9
Sixers	0.6	0.9	0.5	0.4	0.6	0.9	0.5	0.3	0.5	0.6	0.5	0.3	0.1	0.4
Raptors	0.7	0.7	0.8	0.6	0.7	0.6	0.8	0.6	0.7	0.8	1	0.8	0.9	0.3
Bulls	0.7	0.8	0.7	0.4	0.6	0.3	0.6	0.9	0.5	0.5	1	0.9	1	0.8
Cavs	0.4	1	0.4	0.6	1	0.8	1	0.7	1	0.8	1	0.2	0.8	0.8
Pistons	0.3	0.5	0.2	0.4	0.3	0.3	0.3	0.5	0.4	0.3	0.2	0.8	0.2	0.2
Pacers	0.3	0.5	0.6	0.9	0.5	0.5	0.5	0.3	0.4	0.5	0.3	0.4	0.5	0.3
Bucks	0.5	0.6	0.4	0.7	0.7	0.8	0.7	0.2	0.6	0.7	0.2	0.1	0.5	0.5
Hawks	0.2	0.4	0.7	0.7	0.5	0.5	0.6	0.3	0.5	0.5	0.2	0.4	0.4	0.3
Hornets	0.2	0.3	0.4	0.5	0.3	0.3	0.3	0.4	0.4	0.3	0.5	0.2	0.3	0.1
Heat	0.4	0.3	0.9	0.4	0.4	0.3	0.4	0.7	0.4	0.4	0.8	1	0.8	0.7
Magic	0.4	0.2	0.8	0.4	0.2	0.4	0.2	0.4	0.4	0.1	0.6	0.7	0.2	0.5
Wizards	1	0.7	0.7	0.5	0.7	0.6	0.7	0.5	0.6	0.7	0.4	0.3	0.3	0.5
Mavs	0.5	0.4	1	0.5	0.3	0.2	0.3	0.7	0.4	0.4	0.9	0.8	0.7	0.4
Rockets	0.5	0.7	0.7	0.5	0.7	0.6	0.8	0.6	0.7	0.7	0.4	0.5	0.6	0.8
Grizzlies	0.2	0.8	0.4	0.2	0.6	0.4	0.7	0.3	0.6	0.6	0.3	0.4	0.4	0.7
Pelicans	0.2	0.3	0.3	0.4	0.6	0.7	0.4	0.1	0.4	0.6	0.3	0.1	0.5	0.2
Spurs	0.5	0.7	0.6	0.5	0.8	0.7	0.9	0.6	0.8	0.7	0.6	0.4	0.9	0.7
Nuggets	0.7	0.1	0.5	0.9	0.2	0.3	0.2	0.4	0.4	0.3	0.1	0.6	0.2	0.3
Wolves	0.7	0.3	0.3	0.4	0.4	0.8	0.3	0.2	0.4	0.5	0.1	0.1	0.7	0.5
Thunder	0.4	0.6	0.5	0.6	0.5	0.6	0.6	0.6	0.6	0.7	0.6	0.3	0.8	0.4
Blazers	0.4	0.6	0.8	1	0.6	0.5	0.5	0.5	0.5	0.6	0.8	0.7	0.6	0.4
Jazz	0.1	0.4	0.8	0.4	0.4	0.3	0.5	0.3	0.6	0.5	0.9	0.6	0.3	0.5
Warriors	0.8	1	0.7	0.1	1	0.8	1	0.8	1	0.8	0.8	0.5	1	0.9
Clippers	0.8	0.6	0.8	0.1	0.6	0.3	0.6	0.8	0.5	0.2	0.7	0.2	0.6	0.7
Lakers	0.8	0.8	0.8	0.1	0.3	0.5	0.3	0.9	0.4	0.3	0.7	0.9	1	0.8
Suns	0.1	0.4	0.6	0.6	0.4	0.7	0.4	0.6	0.4	0.8	0.4	0.7	0.4	0.3
Kings	0.8	0.2	0.8	0.1	0.3	0.3	0.3	0.4	0.4	0.4	0.5	0.6	0.1	0.2

From this matrix, for each criteria c_k a matrix $\mathcal{O}_k$ is built. As explained in section 5.1., the boolean matrices $\mathcal{O}_k = (b_{ij})$ are built as follows:

$$\beta_{ij} = \begin{cases} 0 & , \ \rho_{ik} < \rho_{jk} \\ 1 & , \ \rho_{ik} \geq \rho_{jk} \end{cases}.$$

Remark 6.2. None of the 14 matrices are detailed here as this would unnecessarily extend the length of this paper.

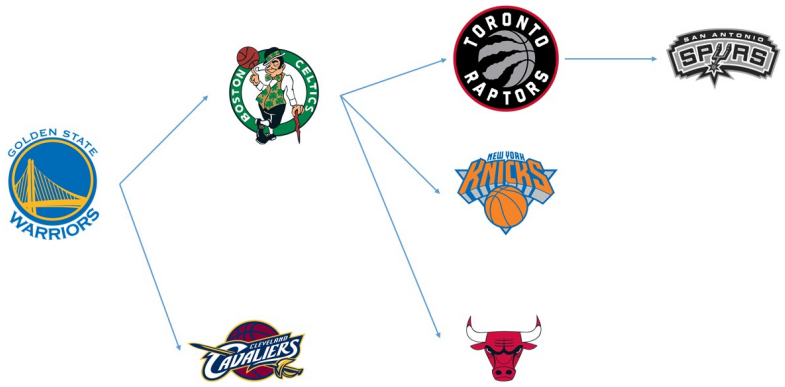
The result of adding these matrices $\mathcal{O}_k$ and dividing every component by 14 (the number of criteria) is the matrix $\mathcal{O}_F$, which shows how much each franchise is preferred over the others:

$$\mathcal{O}_F = \frac{1}{14} \sum_{k=1}^{14} \mathcal{O}_k$$

To avoid operating with matrices of such dimensions, a programming code has been created to apply the algorithm. The programming code can be found in the Appendix, under the title **Programmed Malgrange algorithm**.
The resulting matrix $\mathcal{O}_F$ is in the Appendix in the **Fuzzy preferences matrix** section, as it was too large too show it in portrait mode. Once the matrix showing the degree of preference between teams is built, an α-cut has to be set.
This α-cut will determine from which value on we consider that the degree of preference is strong enough to consider that one team is preferable to another. α is determined at 0.65, with this high value being applied in order that only teams with sufficiently high preference value over others are considered.
Recall that the way of building the boolean matrix of preferences $\mathcal{O}_B$ is by giving a 1 to all the boxes with a higher value than or equal to α, and a 0 otherwise. Again, the matrix $\mathcal{O}_B$ is in the Appendix in the **Boolean preferences matrix** section, as it was also too large to show it in portrait mode. The following step would be to calculate the transitive closing for each team. However, as the purpose of this ordering process is only to find the best team to invest in, the full process will not be applied and interest will only be focused on the top teams.
Therefore, considering the matrix above, it is quite clear that Celtics, Knicks, Raptors, Bulls, Cavs, Spurs and Warriors are the best franchises to invest in. The matrix below is a submatrix of $\mathcal{O}_B$ showing only the preferences between these top seven teams.

	Celtics	*Knicks*	*Raptors*	*Bulls*	*Cavs*	*Spurs*	*Warriors*
Celtics	X	1	1	1	0	1	0
Knicks	0	X	0	0	0	0	0
Raptors	0	0	X	0	0	1	0
Bulls	0	0	0	X	0	0	0
Cavs	0	0	0	0	X	0	0
Spurs	0	0	0	0	0	X	0
Warriors	1	0	0	0	1	1	X

The following graph detailing these preferences will be used to order these 7 franchises:



Looking at the scheme above, it is clear that the Golden State Warriors are the best team to invest in.
The investors then studied the option of buying the Golden State Warriors. However, the Warriors are currently by far the most difficult franchise in the NBA to buy. Its franchise value is around US$2.6 billion, and given its current extraordinary growth the owners are closed to offers.
Therefore, acquiring the Boston Celtics becomes the first option. The Boston Celtics are also a franchise with a high value (US$2.2 billion[1]), and with a well-known history. The idea of acquiring this franchise is not as crazy as that of buying the Warriors, so the investors seriously consider the operation.

Draft workout

Introduction to the NBA Draft

The NBA Draft is an annual event where the 30 franchises choose players from other basketball leagues or from college basketball teams to join the NBA.
Every team chooses two players,resulting in 60 picks (this is the common term used to refer to the elections in the NBA Draft).
The elections are made following an order; one franchise which has the 1st pick, which means that the franchise can choose any eligible player. Following the 1st pick, another franchise has the 2nd pick, and so on. Once the 30th pick is reached, the thirty teams choose again in the same order.
In the 2017-18 Draft, the Boston Celtics are the first team to choose (in NBA argot, the common way to say it would be that the Celtics have the 1st pick in the Draft). That means they can choose any player they want from the ones who have declared themselves eligible for this year's Draft.

[1]Information about franchises valuation has been obtained from Forbes.com; [11] in the Bibliography.

Planning of the Draft workouts

NBA teams plan workouts to see how players perform. These workouts consist of a training session involving some players (usually between 2 and 6) and testing their abilities. These workouts will enable the franchises to decide which players to choose in the NBA Draft.
In this section of the analysis the Draft workouts will be planned. First, by the creation of groups of players who will train together, and then by the assignation of coaches to each group of players.
To find the groups of players the Pichat algorithm will be used. By applying this, groups of similar players will be found, and from here the training groups will be established.
Consider the Top-12 prospects for the 2017-18 NBA Draft, and some key qualities for a future NBA player:

Top12 prospects:	***Qualities:***
p1. Markelle Fultz	q1. Athleticism
p2. Lonzo Ball	q2. Size
p3. Josh Jackson	q3. Defense
p4. De' Aaron Fox	q4. Strength
p5. Jayson Tatum	q5. Quickness
p6. Jonathan Isaac	q6. Leadership
p7. Dennis Smith	q7. Jump Shot
p8. Malik Monk	q8. NBA Ready
p9. Frank Ntilikina	q9. Ball Handling
p10. Zach Collins	q10. Potential
p11. Lauri Markkanen	q11. Passing
p12. Jarrett Allen	q12. Intangibles

According to the opinion of the experts of NBADraft.net, one of the most important websites for NBA prospects, the following scorings for each player have been assigned:

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9	q_{10}	q_{11}	q_{12}
Fultz	0.8	0.8	0.6	0.6	0.8	0.4	0.8	0.6	0.6	0.8	0.6	0.6
Ball	0.6	0.8	0.6	0.4	0.6	0.8	0.4	0.6	0.6	0.8	1	0.6
Jackson	1	0.8	0.8	0.6	0.8	0.6	0.2	0.6	0.4	0.8	0.6	0.8
Fox	0.8	0.6	0.8	0.4	1	0.6	0.4	0.4	0.6	0.8	0.6	0.6
Tatum	0.6	0.6	0.6	0.4	0.4	0.6	0.6	0.4	0.6	0.8	0.6	0.6
Isaac	0.8	0.8	0.6	0.4	0.6	0.6	0.8	0.4	0.8	0.8	0.4	0.6
Smith	1	0.6	0.4	0.6	0.8	0.6	0.4	0.6	0.6	0.8	0.6	0.4
Monk	0.8	0.4	0.4	0.4	0.8	0.4	0.8	0.6	0.4	0.6	0.4	0.6
Ntilikina	0.6	0.8	0.6	0.4	0.6	0.4	0.6	0.4	0.6	0.6	0.6	0.6
Collins	0.6	0.8	0.6	0.6	0.6	0.4	0.6	0.2	0.6	0.8	0.6	0.6
Markkanen	0.4	0.8	0.4	0.4	0.4	0.4	0.8	0.6	0.6	0.6	0.6	0.6
Allen	0.6	0.6	0.4	0.4	0.6	0.6	0.4	0.4	0.6	0.8	0.4	0.6

Remark 6.3. NBADraft.net uses a scale between 1 and 10 for the qualities. As all these 12 players have a scoring higher than or equal to 6 for each quality, the following mapping has been done:

NBADraft.net scoring	***0 to 1 scale***
6	0.2
7	0.4
8	0.6
9	0.8
10	1

At this point, two more steps are required to begin applying the Pichat algorithm:

I. Apply an α-cut to create a boolean matrix which shows which players have the quality and which do not. As the values in the fuzzy matrix are quite high, we set an α-cut equal to 0.8. The following boolean matrix results:

	q_1	q_2	q_3	q_4	q_5	q_6	q_7	q_8	q_9	q_{10}	q_{11}	q_{12}
Fultz	1	1	0	0	1	0	1	0	0	1	0	0
Ball	0	1	0	0	0	1	0	0	0	1	1	0
Jackson	1	1	1	0	1	0	0	0	0	1	0	1
Fox	1	0	1	0	1	0	0	0	0	1	0	0
Tatum	0	0	0	0	0	0	0	0	0	1	0	0
Isaac	1	1	0	0	0	0	1	0	1	1	0	0
Smith	1	0	0	0	1	0	0	0	0	1	0	0
Monk	1	0	0	0	1	0	1	0	0	0	0	0
Ntilikina	0	1	0	0	0	0	0	0	0	0	0	0
Collins	0	1	0	0	0	0	0	0	0	1	0	0
Markkanen	0	1	0	0	0	0	1	0	0	0	0	0
Allen	0	0	0	0	0	0	0	0	0	1	0	0

The way to understand this matrix is that, for example, Fultz has the qualities $q_1, q_2, q_5, q_7, q_{10}$, but not the qualities $q_3, q_4, q_6, q_8, q_9, q_{11}, q_{12}$.

II. From the matrix obtained in step I., create the similarities matrix. First, the Hamming distance for every pair of players has to be calculated.

$$d(p_i, p_j) = \frac{1}{12} \sum_{k=1}^{12} \mid m_{ik} - m_{jk} \mid,$$

where m_{xy} are the elements from the fuzzy matrix prior to the α-cut in Step I.
Once the Hamming distances are calculated for every pair of players, a matrix with the Hamming distances is built.[2] Hamming distance measures the size of the difference between two players. The higher the distance, the bigger the difference. Then, if we want to measure similarity between players, we have to find the complementary of the distances, understanding the complementary as $1 - d(p_i, p_j)$, for each pair of players p_i,p_j.

[2]The matrix with the Hamming distances can be found in the Appendix, in the Draft workouts section.

The following matrix shows the similarities for the 12 players:

	Fultz	*Ball*	*Jackson*	*Fox*	*Tatum*	*Isaac*	*Smith*	*Monk*	*Ntilikina*	*Collins*	*Markannen*	*Allen*
Fultz	1	0.85	0.87	0.87	0.87	0.90	0.88	0.88	0.90	0.92	0.88	0.83
Ball	0.85	1	0.82	0.85	0.88	0.85	0.83	0.77	0.88	0.87	0.83	0.88
Jackson	0.87	0.82	1	0.87	0.80	0.80	0.88	0.78	0.80	0.82	0.75	0.80
Fox	0.87	0.85	0.87	1	0.90	0.87	0.88	0.82	0.87	0.85	0.78	0.90
Tatum	0.87	0.88	0.80	0.90	1	0.90	0.85	0.82	0.93	0.92	0.88	0.93
Isaac	0.90	0.85	0.80	0.87	0.90	1	0.82	0.85	0.90	0.88	0.85	0.90
Smith	0.88	0.83	0.88	0.88	0.85	0.82	1	0.83	0.82	0.83	0.80	0.88
Monk	0.88	0.77	0.78	0.82	0.82	0.85	0.83	1	0.85	0.80	0.87	0.85
Ntilikina	0.90	0.88	0.80	0.87	0.93	0.90	0.82	0.85	1	0.95	0.92	0.90
Collins	0.92	0.87	0.82	0.85	0.92	0.88	0.83	0.80	0.95	1	0.87	0.88
Markannen	0.88	0.83	0.75	0.78	0.88	0.85	0.80	0.87	0.92	0.87	1	0.85
Allen	0.83	0.88	0.80	0.90	0.93	0.90	0.88	0.85	0.90	0.88	0.85	1

To begin applying the Pichat algorithm, it is only necessary to determine the value α from which two players are considered similar. Taking a first look at the matrix, it is easy to see that the similarity scorings are considerably high, so applying a low α-cut would mean considering almost all players as being similar, and therefore the grouping would not be as useful as required.

The α-cut chosen is 0.88. That means that any two players with a similarity scoring higher than or equal to 0.88 are similar, and if their similarity valuation is lower than 0.88 they are not considered similar.

Note that the elements below the diagonal can be ignored, as they give repeated information because the matrix is symmetric (and, therefore, the similarity between p_i and p_j is the same as the similarity between p_j and p_i). The similarity matrix after applying the α-cut is:

	Fultz	*Ball*	*Jackson*	*Fox*	*Tatum*	*Isaac*	*Smith*	*Monk*	*Ntilikina*	*Collins*	*Markannen*	*Allen*
Fultz	1	0	0	0	0	1	1	1	1	1	1	0
Ball		1	0	0	1	0	0	0	1	0	0	1
Jackson			1	0	0	0	1	0	0	0	0	0
Fox				1	1	0	1	0	0	0	0	1
Tatum					1	1	0	0	1	1	1	1
Isaac						1	0	0	1	1	0	1
Smith							1	0	0	0	0	1
Monk								1	0	0	0	0
Ntilikina									1	1	1	1
Collins										1	0	1
Markannen											1	0
Allen												1

Now, the Pichat algorithm is applied (the algorithm is detailed in section 4.4).
The elements of the boolean addition are shown in the following table:

Row	***Sum elements***
1	1+2,3,4,5,12
2	2+3,4,6,7,8,10,11
3	3+4,5,6,8,9,10,11,12
4	4+6,8,9,10,11
5	5+7,8
6	6+7,8,11
7	7+8,9,10,11
8	8+9,10,11,12
9	-
10	10+11
11	11+12
12	-

Calculating the boolean addition, the following result is obtained:

$$\mathcal{P} = 1,2,3,4,6,7,8,10,12 + 1,2,3,4,7,8,11 + 1,2,3,5,6,8,9,10,11 + 1,2,3,6,7,8,9,10,11 + 1,2,4,5,6,8,9,10,11,12 + \\ + 1,3,4,6,8,10,11 + 2,3,4,5,6,7,8,10,12 + 2,3,4,5,6,7,9,10,11,12 + 2,3,4,5,6,8,9,10,11,12 + \\ + 2,3,4,5,7,8,11,12 + 2,3,4,6,7,8,10,11$$

Remark 6.4. The calculation of the addition is detailed in the section ***Pichat algorithm calculation*** in the Appendix.

Calculating the complementary of each term in $\mathcal{P}$ the maximal similarity subrelations are found:

$$\{5,9,11\}, \{5,6,9,10,12\}, \{4,7,12\}, \{4,5,12\}, \{3,7\}, \{2,5,7,9,12\}, \{1,9,11\}, \{1,8\}, \{1,7\}, \{1,6,9,10\}, \{1,5,9,12\}$$

From these similarity relations, the groups for the draft workouts have to be determined:

- Players 6, 10 and 12 are very similar according to the result obtained with Pichat. Player 1 is also quite similar to these three players, as they are together in different maximal subrelations. Therefore, players 1, 6, 10 and 12 will work out together.
- Players 5, 9 and 11 are very similar. In fact, they have a maximal subrelation. Player 2 also shares similarities with players 9 and 11 (and only appears in one maximal subrelation). Therefore we consider players 2, 5, 9 and 11 as another training group.

- The four players left are not very similar to any of the players in the list, this is why they have been left with no group until the end. They are not similar to be able to train together, so the best option is to split them into two pairs. Players 4 and 7 share a maximal similarity subrelation, so they will train together.

- Players 3 and 8, despite sharing very few qualities, will form the fourth training group.

Therefore, the four groups established are:

$$G_1 = \{1, 6, 10, 12\}, G_2 = \{2, 5, 9, 11\}, G_3 = \{3, 8\}, G_4 = \{4, 7\}.$$

Using players names:

$$G_1 = \{\text{Fultz, Isaac, Collins, Allen}\},\ G_2 = \{\text{Ball, Tatum, Ntilikina, Markannen}\}\ ,$$
$$G_3 = \{\text{Jackson, Monk}\},\ G_4 = \{\text{Fox, Smith}\}.$$

Costs minimization

After grouping the players, five leading coaches are contacted and asked for cost estimates for specific workouts for these groups of players.
The costs of each coach (taking into account dates, accomodation, additional training staff, etc.) are shown in the following table:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 15,000 & 15,500 & 15,000 & 16,500 & 16,500 \\ 14,000 & 15,500 & 15,000 & 15,500 & 13,500 \\ 10,000 & 11,000 & 12,000 & 12,500 & 11,000 \\ 10,000 & 10,500 & 11,000 & 11,500 & 9,500 \end{pmatrix} \end{array}$$

Applying the Hungarian algorithm, the aim is to find which combination minimizes the costs. Hungarian algorithm only applies on square matrices, therefore a ficticious group of players (which will be called g_F to make clear that it is fictitious) will be added to the matrix. This fictitious row will be assigned the maximum value in the matrix:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_F \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 15,000 & 15,500 & 15,000 & 16,500 & 16,500 \\ 14,000 & 15,500 & 15,000 & 15,500 & 13,500 \\ 10,000 & 11,000 & 12,000 & 12,500 & 11,000 \\ 10,000 & 10,500 & 11,000 & 11,500 & 9,500 \\ 16,500 & 16,500 & 16,500 & 16,500 & 16,500 \end{pmatrix} \end{array}$$

Dividing each element by the maximum value in the matrix; i.e., 16,500, a matrix with values between 0 and 1 is obtained:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_F \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 0.9394 & 0.9394 & 0.9091 & 1 & 1 \\ 0.8485 & 0.9394 & 0.9091 & 0.9394 & 0.8182 \\ 0.6061 & 0.6667 & 0.7273 & 0.7576 & 0.6667 \\ 0.6061 & 0.6364 & 0.6667 & 0.6970 & 0.5758 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

Applying the Hungarian algorithm to the matrix above, the following optimal assignation is obtained.

Group 1 $\rightarrow$ Coach 3

Group 2 $\rightarrow$ Coach 5

Group 3 $\rightarrow$ Coach 1

Group 4 $\rightarrow$ Coach 2

Group F $\rightarrow$ Coach 4

Remark 6.5. There are four groups of players, $\{G_1, G_2, G_3, G_4\}$, and five coaches,$\{c_1, c_2, c_3, c_4, c_5\}$, therefore is trivial that one of the coaches will not have any group of players assigned. The results obtained using the algorithm show that Coach 4 has been assigned to the fictitious group of players; this means that Coach 4 is the one who will not train any of the groups.

Detail of the calculation is given the Appendix, in the Hungarian algorithm application section.

Marketing actions

The intention of the investors is to boost the franchise to the next level. With this purpose, the board has organised a brainstorming with some marketing actions that can help to update the image of the franchise.

s1. Create a new logo and new jersey design.

s2. Create a new image on Twitter, Instagram and other social networks.

s3. Hire influencer-services for ad campaigns.

s4. Collaborate with NGOs.

s5. Sign a Chinese player. (add some explanation)

s6. Organize an amateur 3x3 tournament at the home arena.

s7. Find a sponsor for the jersey.

s8. Open stores worldwide selling the franchise merchandise.

s9. Develop an own mobile app.

s10. Offer reduced-prize season tickets for senior citizens.

s11. Organize a preseason European tour.

The expected results of these marketing actions are as follows:

r1. An increase in merchandise sales.

r2. Stronger connection with young fans.

r3. Improved team image.

r4. Global recognition of the franchise image and brand awareness.

r5. A feeling of community among fans.

Once the actions have been suggested, and the objectives are clear, the board decides to apply the forgotten effects algorithm to analyze the interrelations between actions and effects. By applying this method, some indirect effects will come to light and therefore we will be able to know if some of the marketing actions planned during the brainstorming have some unexpected repercussions.
The way to proceed is exactly as that explained in section 2.4. Therefore, the three matrices necessary to perform the Forgotten Effects algorithm are built, according to the opinion of the board:

- Matrix $M_{\mathcal{R}}$:

$$\begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \\ s_{11} \end{array} \begin{array}{c} \begin{array}{ccccc} r_1 & r_2 & r_3 & r_4 & r_5 \end{array} \\ \begin{pmatrix} 0.9 & 0.8 & 0 & 0.6 & 0.2 \\ 0.9 & 0.1 & 0.2 & 0.9 & 0.6 \\ 0.8 & 0.9 & 0.3 & 0.4 & 0 \\ 0 & 0.2 & 1 & 0.1 & 0.1 \\ 0.9 & 0.5 & 0.4 & 1 & 0 \\ 0.3 & 0.6 & 0.8 & 0 & 0.9 \\ 0.6 & 0.5 & 0 & 0.3 & 0 \\ 0.4 & 0.2 & 0.2 & 0.9 & 0 \\ 0.1 & 0.6 & 0.3 & 0.2 & 0.6 \\ 0 & 0 & 0.8 & 0 & 0.4 \\ 0.4 & 0.4 & 0.3 & 0.8 & 0.2 \end{pmatrix} \end{array}$$

- Matrix M_X:

$$\begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \\ s_{11} \end{array} \begin{array}{c} \begin{array}{ccccccccccc} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 & s_7 & s_8 & s_9 & s_{10} & s_{11} \end{array} \\ \begin{pmatrix} 1 & 0.2 & 0.6 & 0 & 0.3 & 0 & 0.7 & 0.9 & 0.4 & 0 & 0 \\ 0.7 & 1 & 0.9 & 0.2 & 0.6 & 0.3 & 0.2 & 0.8 & 0.9 & 0.1 & 0.3 \\ 0.8 & 1 & 1 & 0.2 & 0.4 & 0.7 & 0 & 0.3 & 0.7 & 0.1 & 0.3 \\ 0 & 0.3 & 0.2 & 1 & 0.7 & 0.3 & 0 & 0.3 & 0 & 0.4 & 0.2 \\ 0.7 & 0.8 & 0.9 & 0.2 & 1 & 0 & 0.6 & 0.8 & 0.7 & 0 & 0.2 \\ 0.2 & 0.4 & 0.3 & 0.5 & 0 & 1 & 0 & 0 & 0 & 0.2 & 0 \\ 0.9 & 0.9 & 0.7 & 0 & 0.4 & 0 & 1 & 0 & 0.3 & 0.4 & 0.5 \\ 0.3 & 0.8 & 0.9 & 0.2 & 0.5 & 0 & 0.4 & 1 & 0 & 0 & 0.6 \\ 0 & 0.8 & 0.4 & 0 & 0.7 & 0.1 & 0.3 & 0.3 & 1 & 0.1 & 0.4 \\ 0 & 0.2 & 0.2 & 0 & 0 & 0 & 0 & 0 & 0.2 & 1 & 0 \\ 0 & 0.8 & 0.5 & 0.4 & 0.2 & 0 & 0.2 & 0.3 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

- Matrix M_Y:

$$\begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \end{array} \begin{array}{c} \begin{array}{ccccc} r_1 & r_2 & r_3 & r_4 & r_5 \end{array} \\ \begin{pmatrix} 1 & 0.4 & 0.2 & 0.3 & 0 \\ 0.9 & 1 & 0.1 & 0.1 & 0.3 \\ 0.3 & 0.4 & 1 & 0.4 & 0.2 \\ 0.9 & 0.6 & 0.4 & 1 & 0 \\ 0.2 & 0.4 & 0.5 & 0 & 1 \end{pmatrix} \end{array}$$

To avoid operating with large dimension matrices, a programming code has been created to apply the algorithm. The programming code is in the Appendix, under the title of **Programmed forgotten effects algorithm**. After running the programming code, the following result is obtained:

$$\begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \\ s_8 \\ s_9 \\ s_{10} \\ s_{11} \end{array} \begin{array}{c} \begin{array}{ccccc} r_1 & r_2 & r_3 & r_4 & r_5 \end{array} \\ \begin{pmatrix} 0 & 0 & 0.4 & 0.3 & 0.2 \\ 0 & 0 & 0.3 & 0 & 0 \\ 0.1 & 0 & 0.4 & 0.5 & 0.7 \\ 0.7 & 0.4 & 0 & 0.6 & 0.3 \\ 0 & 0.4 & 0.1 & 0 & 0.6 \\ 0.3 & 0 & 0 & 0.4 & 0 \\ 0.3 & 0.3 & 0.5 & 0.6 & 0.6 \\ 0.5 & 0.7 & 0.3 & 0 & 0.6 \\ 0.7 & 0 & 0.2 & 0.6 & 0 \\ 0.3 & 0.4 & 0 & 0.4 & 0 \\ 0.4 & 0.2 & 0.2 & 0 & 0.4 \end{pmatrix} \end{array}$$

At first glance, the indirect relations with a higher scoring (of 0.7) can be highlighted:

Relation	***Action***	***Effect***
$s_3 - r_5$	Hire influencer-services for ad campaigns	A feeling of community between fans
$s_4 - r_1$	Collaborate with NGOs	An increase in merchandise sales
$s_8 - r_2$	Open stores worldwide selling the franchise merchandise	Stronger connection with young fans
$s_9 - r_1$	Develop an own mobile app	An increase in merchandise sales

This means that some marketing actions will have more relation than expected with some of the consequences. For example, collaborating with NGOs can have an impact on merchandise sales. The relation between these, which at first seemed almost insignificant, is very strong. In light of these results, some of the actions to take may be reconsidered.

Conclusions

The theory of decision comprises and studies the basic elements in the decision-making process. From an analytic point of view, but at the same time providing a high degree of flexibility, the theory of decision amis to cope with problems where uncertainty is present. After working on the project, I consider that one of the most differentiating aspects of the theory of decision, and consequently of the algorithms presented, is the wide variety of problems that can be addressed. This versatility is demonstrated in the application case, where an investment analysis, a creation of training groups, a cost minimization problem and a marketing study have been carried out employing different algorithms pertaining to the theory of decision.

All the algorithms are based on fuzzy mathematics. The data treatment given by fuzzy mathematics brings the possibility of adding subjectivity to the analysis, which classical mathematics cannot do. Alpha-cuts offer the possibility to consider different scenarios in the decision-making process depending on the exigence level imposed. Subjectivity plays an important role in fuzzy mathematics, and therefore the same problem analyzed by two different individuals can, and in fact probably will, have two different solutions. Therefore, one of the conclusions that can be drawn is that fuzzy and classical mathematics present varying results to the same problem since each looks at it from a very different point of view.

Apart from being very adaptable even to a very specific case, another benefit of algorithms is their simplicity and ease of application. When the number of dimensions in the problem is considerably large, the difficulty of applying the algorithm "by hand" increases. In this project a solution has been proposed. As algorithms use a simple process, it is easy to program them so they can be applied to problems of any dimension. In this project, the Forgotten Effects method and the process required to start applying the Malgrange algorithm have been programmed.

This project is a theoretical approach to the theory of decision, and as a consequence there are no hypotheses to be either corroborated or rejected. Nevertheless, it has been demonstrated that the theory of decision is useful in many fields. Other business decisions, such as processes involving project viability analysis, investment portfolio management or recruitment processes could also have been analyzed.

As stated in the introduction, the aim of this project is to provide a general overview of the theory of decision and fuzzy mathematics. Further studies can be carried out along the same lines as the project; for example, an in-depth study of algorithms from a mathematical point of view, demonstrating their validity and proving their coherence.

Bibliography

[1] GIL-ALUJA, Jaime, *Elements for a theory of decision in uncertainty*, Milladoiro, (1999).

[2] SERRANG, Oliver. Harvard University. A fast numerical method for max-min convolution and the application to efficient max-product inference in Bayesian networks, January 2015. [Consulted in April 2017.]
Link: http://adsabs.harvard.edu/abs/2015arXiv150102627S

[3] Stanford University. An introduction to Philosophy; the Law of Excluded Middle. [Consulted in April 2017].
Link: https://web.stanford.edu/ bobonich/glances%20ahead/IV.excluded.middle.html

[4] KEEFER, Julia L. New York University. Traditional Logic Versus Fuzzy Thinking. [Consulted in April 2017].
Link: http://www.nyu.edu/classes/keefer/ww1/fuzz.html

[5] Salés Vallès, *F.A Aplicaciones de Galois en conjuntos ordenados y en retículos*, Collectanea Mathematica, (1970)

[6] U.S. Department of Commerce, Bureau of Economic Analysis, March 2017. [Consulted in May 2017].
Link: https://www.bea.gov/newsreleases/regional/spi/2017/pdf/spi0317.pdf

[7] Numbeo.com. North America: Quality of Life Index 2017. [Consulted in May 2017].
Link: https://www.numbeo.com/quality-of-life/region_rankings.jsp?title=2017&region=021

[8] ESPN.com. NBA Attendance Report - 2017. [Consulted in May 2017].
Link: http://www.espn.com/nba/attendance

[9] BleacherReport.com. Ranking All 30 NBA Arenas. [Consulted in May 2017].
Link: http://bleacherreport.com/articles/1374029-ranking-all-30-nba-arenas

[10] Ranker.com. Your Favorite NBA Basketball Teams. [Consulted in May 2017].
Link: http://www.ranker.com/crowdranked-list/favorite-nba-basketball-teams

[11] Badenhausen, Kurt. Forbes.com, The Knicks and Lakers Top the NBA's Most Valuable Teams 2017. [Consulted in May 2017].
Link: https://www.forbes.com/sites/kurtbadenhausen/2017/02/15/the-knicks-and-lakers-head-the-nbas-most-valuable-teams-2017/#291bec9a7966

Appendix

The appendix contains the following sections:

- Criteria evaluation

- Fuzzy preferences matrix

- Boolean preferences matrix

- Draft workouts

- Programming codes for the algorithms

Criteria evaluation

This section of the Appendix gives details about how the scorings to create the matrix to apply the Malgrange algorithm have been assigned.

Criteria 1

To determine the economic situation of the state for every franchise, a ranking from the U.S. Department of Commerce has been employed ([6] in the Bibliography). This ranking creates a list of the states ordered from best to worst according to economic situation. Considering there are 50 states in the USA, the following scoring has been assigned to each state depending on its position in the Department of Commerce ranking:

Top 3	4th-6th	7th-10th	11th-15th	16th-20th	21th-25th
1	0.9	0.8	0.7	0.6	0.5

26th-30th	31st-35th	36th-40th	41st-45th	>46th
0.4	0.3	0.2	0.1	0

Criteria 2

The scoring for the importance of basketball in the city has been calculated by considering the popularity of basketball in relation to other sports such as baseball, hockey and football.

Criteria 3

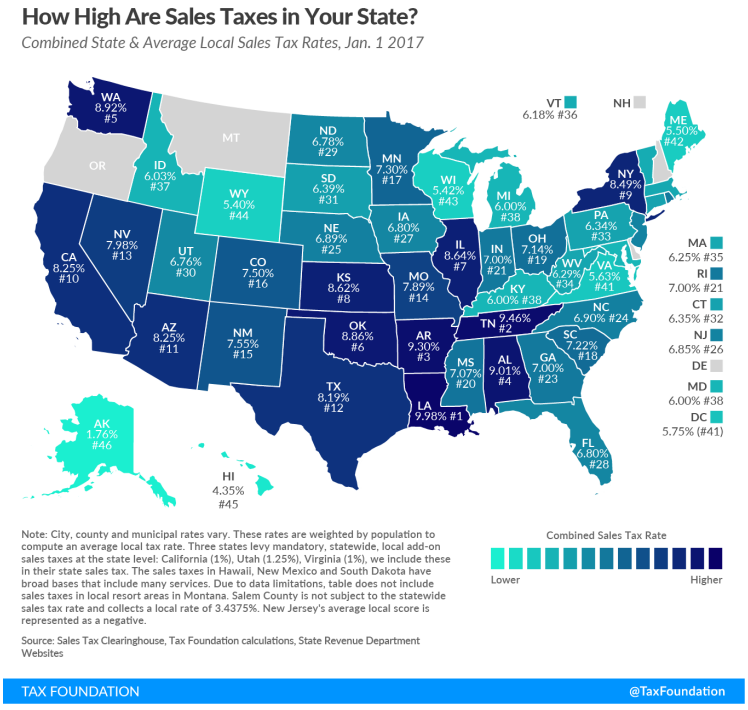
To measure weather and social conditions a ratio has been calculated. This ratio is a weighted average between weather conditions (W) and quality of life (Q):

$$c_3 = 0.3W + 0.7Q$$

The Quality of Life Index ([7] in Bibliography) has been used to measure the variable Q.

Criteria 4

The information to evaluate the scoring for tax regulations has been obtained from the website *www.taxfoundation.org*. The maximum scoring has only been assigned in the case of tax-free state. Otherwise, the scoring is gradual; the higher the taxes, the lower the scoring.



Criteria 5

Evaluating the current situation of each team in order to apply a value between 0 and 1 is quite subjective. The scoring has been assigned depending on last year's results and

team progression; current results, standings and short-term historic performance.

Criteria 6

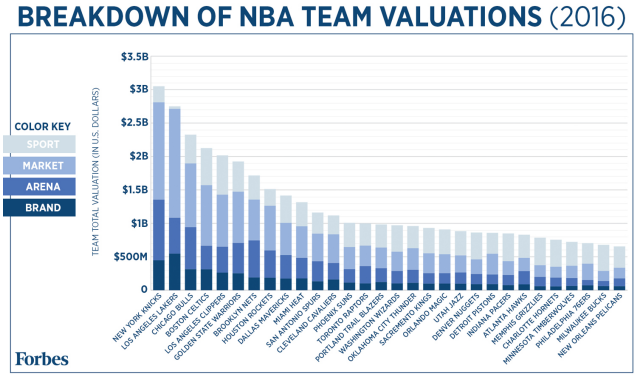
Team projection is even more subjective than the current situation of the team. The scoring for this criteria has been based on the mean age of the roster, the future expectations for each player, team stability, team cohesion and immediate future plans.

Criteria 7

Playoff chances for every franchise have been calculated by projecting the franchise trajectory in the last years together with the expected improvement or deterioration of the team.

Criteria 8

Scoring for marketing income has been determined based on a Forbes study measuring income for each team, and grouping it according to the nature of the income. The graph below shows the results of Forbes study:



Criteria 9

Scoring for media income has been determined based on a Forbes study measuring income for each team, and grouping it according to the nature of the income. The graph shown under Criteria 8 also provides information about media income.

Criteria 10

This criteria makes reference to the stability of the franchise and its board. The way to determine the scoring for this criteria has been to consider the key events that have taken place in each franchise during the last 3 years.

Criteria 11

Scoring for arena attendance has been determined using data obtained from the official website of *ESPN* ([8] in the Bibliography).
The first 3 teams have been given a scoring of 1, the following three a scoring of 0.9, and so forth until the worst three teams in terms of attendance which were given a scoring of 0.1.

Criteria 12

Scoring for arena facilites has been determined following a ranking by the website *Bleacher Report* ([9] in the Bibliography) analyzing the facilities of all the NBA arenas.
The first 3 teams have been given a scoring of 1, the following three a scoring of 0.9, and so forth until the three worst three teams in the ranking which were given a scoring of 0.1.

Criteria 13

Scoring for the popularity of every team in the USA is based on a popular ranking on the website *Ranker.com* ([10] in the Bibliography) where people have been registering their votes since 2014.

Criteria 14

As there are no rankings for popularity of NBA franchises outside the USA, this scoring has been calculated with a high component of subjectivity based on last years experiences and facts.

Fuzzy preferences matrix

$$\mathcal{O}_F = \begin{array}{c} \\ \textit{Celtics} \\ \textit{Nets} \\ \textit{Knicks} \\ \textit{Sixers} \\ \textit{Raptors} \\ \textit{Bulls} \\ \textit{Cavs} \\ \textit{Pistons} \\ \textit{Pacers} \\ \textit{Bucks} \\ \textit{Hawks} \\ \textit{Hornets} \\ \textit{Heat} \\ \textit{Magic} \\ \textit{Wizards} \\ \textit{Mavs} \\ \textit{Rockets} \\ \textit{Grizzlies} \\ \textit{Pelicans} \\ \textit{Spurs} \\ \textit{Nuggets} \\ \textit{Wolves} \\ \textit{Thunder} \\ \textit{Blazers} \\ \textit{Jazz} \\ \textit{Warriors} \\ \textit{Clippers} \\ \textit{Lakers} \\ \textit{Suns} \\ \textit{Kings} \end{array} \begin{array}{c} \begin{array}{cccccccccccccccccccccccccccc} \textit{Celtics} & \textit{Nets} & \textit{Knicks} & \textit{Sixers} & \textit{Raptors} & \textit{Bulls} & \textit{Cavs} & \textit{Pistons} & \textit{Pacers} & \textit{Bucks} & \textit{Hawks} & \textit{Hornets} & \textit{Heat} & \textit{Magic} & \textit{Wizards} & \textit{Mavs} & \textit{Rockets} & \textit{Grizzlies} & \textit{Pelicans} & \textit{Spurs} & \textit{Nuggets} & \textit{Wolves} & \textit{Thunder} & \textit{Blazers} & \textit{Jazz} & \textit{Warriors} & \textit{Clippers} & \textit{Lakers} & \textit{Suns} & \textit{Kings} \end{array} \\ \begin{pmatrix} X & 0.9286 & 0.7143 & 0.9286 & 0.7143 & 0.7143 & 0.5000 & 0.9286 & 0.8571 & 0.8571 & 0.9286 & 0.9286 & 0.7857 & 0.8571 & 0.8571 & 0.7857 & 0.8571 & 0.9286 & 0.9286 & 0.8571 & 0.8571 & 0.9286 & 0.8571 & 0.7143 & 0.7857 & 0.4286 & 0.8571 & 0.7143 & 0.8571 & 0.8571 \\ 0.2857 & X & 0.2143 & 0.5000 & 0.2857 & 0.2857 & 0.3571 & 0.4286 & 0.4286 & 0.4286 & 0.4286 & 0.5714 & 0.3571 & 0.5000 & 0.4286 & 0.2857 & 0.4286 & 0.4286 & 0.5714 & 0.4286 & 0.5000 & 0.6429 & 0.4286 & 0.2143 & 0.2857 & 0.3571 & 0.3571 & 0.2857 & 0.4286 & 0.5000 \\ 0.4286 & 0.8571 & X & 0.7143 & 0.5000 & 0.5714 & 0.4286 & 0.9286 & 0.7143 & 0.7143 & 0.7143 & 0.9286 & 0.7143 & 0.9286 & 0.6429 & 0.7857 & 0.6429 & 0.7143 & 0.7857 & 0.5714 & 0.8571 & 0.8571 & 0.6429 & 0.7143 & 0.8571 & 0.5000 & 0.7857 & 0.8571 & 0.8571 & 0.8571 \\ 0.2143 & 0.5714 & 0.3571 & X & 0.1429 & 0.4286 & 0.2857 & 0.7143 & 0.6429 & 0.5000 & 0.5714 & 0.7143 & 0.5714 & 0.5714 & 0.2857 & 0.5000 & 0.2857 & 0.6429 & 0.8571 & 0.2143 & 0.5714 & 0.7857 & 0.4286 & 0.5000 & 0.5714 & 0.1429 & 0.5000 & 0.5000 & 0.5000 & 0.6429 \\ 0.3571 & 0.7857 & 0.5714 & 0.8571 & X & 0.6429 & 0.5714 & 0.9286 & 0.8571 & 0.8571 & 0.8571 & 0.9286 & 0.7143 & 0.9286 & 0.8571 & 0.7857 & 0.9286 & 0.8571 & 0.9286 & 0.7143 & 0.8571 & 0.9286 & 0.9286 & 0.8571 & 0.9286 & 0.4286 & 0.7857 & 0.5714 & 0.9286 & 0.8571 \\ 0.3571 & 0.7857 & 0.5000 & 0.7857 & 0.5000 & X & 0.5000 & 0.9286 & 0.8571 & 0.5714 & 0.8571 & 0.8571 & 0.7857 & 0.8571 & 0.5000 & 0.7857 & 0.5714 & 0.7143 & 0.8571 & 0.5714 & 0.8571 & 0.9286 & 0.7143 & 0.7143 & 0.7857 & 0.5000 & 0.7857 & 0.7857 & 0.7857 & 0.7857 \\ 0.5000 & 0.6429 & 0.5714 & 0.7143 & 0.6429 & 0.5714 & X & 0.8571 & 0.7143 & 0.7857 & 0.7143 & 0.9286 & 0.7857 & 0.7857 & 0.7143 & 0.7143 & 0.7143 & 0.8571 & 0.9286 & 0.6429 & 0.6429 & 0.8571 & 0.7857 & 0.7143 & 0.7857 & 0.5714 & 0.7143 & 0.5714 & 0.7857 & 0.7143 \\ 0.0714 & 0.6429 & 0.1429 & 0.3571 & 0.1429 & 0.1429 & 0.1429 & X & 0.4286 & 0.2857 & 0.4286 & 0.6429 & 0.2857 & 0.7143 & 0.2143 & 0.4286 & 0.1429 & 0.3571 & 0.5000 & 0.1429 & 0.7143 & 0.5714 & 0.1429 & 0.2143 & 0.4286 & 0.2143 & 0.2857 & 0.4286 & 0.3571 & 0.6429 \\ 0.1429 & 0.6429 & 0.3571 & 0.5000 & 0.1429 & 0.2143 & 0.2857 & 0.7857 & X & 0.5000 & 0.7143 & 0.7857 & 0.5000 & 0.5714 & 0.2857 & 0.5000 & 0.1429 & 0.5714 & 0.7857 & 0.2857 & 0.7143 & 0.7857 & 0.3571 & 0.1429 & 0.6429 & 0.1429 & 0.2857 & 0.4286 & 0.6429 & 0.5714 \\ 0.1429 & 0.5714 & 0.3571 & 0.5000 & 0.2143 & 0.4286 & 0.2857 & 0.7857 & 0.5714 & X & 0.7143 & 0.7143 & 0.5714 & 0.6429 & 0.5000 & 0.5714 & 0.3571 & 0.6429 & 0.8571 & 0.2857 & 0.5714 & 0.7857 & 0.5714 & 0.5000 & 0.6429 & 0.1429 & 0.5000 & 0.4286 & 0.5714 & 0.5714 \\ 0.1429 & 0.6429 & 0.2857 & 0.5714 & 0.1429 & 0.4286 & 0.2857 & 0.6429 & 0.5714 & 0.4286 & X & 0.7857 & 0.5000 & 0.5714 & 0.3571 & 0.5000 & 0.2143 & 0.5000 & 0.6429 & 0.2857 & 0.6429 & 0.7857 & 0.4286 & 0.2143 & 0.6429 & 0.2143 & 0.4286 & 0.4286 & 0.6429 & 0.5714 \\ 0.0714 & 0.5714 & 0.1429 & 0.3571 & 0.0714 & 0.1429 & 0.2143 & 0.6429 & 0.2857 & 0.3571 & 0.2857 & X & 0.2857 & 0.6429 & 0.2857 & 0.3571 & 0.2143 & 0.4286 & 0.6429 & 0.1429 & 0.6429 & 0.6429 & 0.0714 & 0.0714 & 0.3571 & 0.1429 & 0.2857 & 0.4286 & 0.2857 & 0.6429 \\ 0.2857 & 0.7857 & 0.4286 & 0.5000 & 0.2857 & 0.2857 & 0.4286 & 0.8571 & 0.5714 & 0.4286 & 0.5000 & 0.8571 & X & 0.9286 & 0.4286 & 0.5714 & 0.4286 & 0.5714 & 0.7857 & 0.3571 & 0.7857 & 0.7857 & 0.5000 & 0.5000 & 0.5714 & 0.3571 & 0.5000 & 0.6429 & 0.7143 & 0.8571 \\ 0.1429 & 0.5714 & 0.1429 & 0.5000 & 0.1429 & 0.2143 & 0.2857 & 0.5000 & 0.5000 & 0.3571 & 0.4286 & 0.5000 & 0.2857 & X & 0.2857 & 0.1429 & 0.2857 & 0.5000 & 0.5714 & 0.2857 & 0.7143 & 0.5000 & 0.3571 & 0.2857 & 0.4286 & 0.2857 & 0.2857 & 0.2857 & 0.4286 & 0.6429 \\ 0.2857 & 0.7143 & 0.4286 & 0.7857 & 0.2857 & 0.5714 & 0.2857 & 0.8571 & 0.7143 & 0.7857 & 0.7143 & 0.8571 & 0.5714 & 0.7143 & X & 0.5714 & 0.5714 & 0.7143 & 0.8571 & 0.4286 & 0.7857 & 0.8571 & 0.6429 & 0.5714 & 0.7143 & 0.2857 & 0.6429 & 0.5000 & 0.5714 & 0.7143 \\ 0.2143 & 0.7857 & 0.3571 & 0.5000 & 0.2857 & 0.2143 & 0.3571 & 0.8571 & 0.5714 & 0.5000 & 0.5714 & 0.9286 & 0.6429 & 0.9286 & 0.5000 & X & 0.5714 & 0.5714 & 0.7143 & 0.5000 & 0.7857 & 0.7143 & 0.4286 & 0.5000 & 0.6429 & 0.3571 & 0.5000 & 0.5714 & 0.6429 & 0.8571 \\ 0.2143 & 0.7857 & 0.3571 & 0.7143 & 0.4286 & 0.5000 & 0.2857 & 0.8571 & 0.8571 & 0.8571 & 0.8571 & 0.8571 & 0.5714 & 0.7143 & 0.8571 & 0.5714 & X & 0.8571 & 0.9286 & 0.5714 & 0.7143 & 0.7857 & 0.7143 & 0.6429 & 0.7143 & 0.2857 & 0.6429 & 0.4286 & 0.7143 & 0.6429 \\ 0.1429 & 0.5714 & 0.4286 & 0.5714 & 0.1429 & 0.4286 & 0.2143 & 0.6429 & 0.6429 & 0.5714 & 0.7857 & 0.7143 & 0.4286 & 0.5000 & 0.4286 & 0.4286 & 0.1429 & X & 0.7857 & 0.2143 & 0.5714 & 0.7143 & 0.4286 & 0.4286 & 0.6429 & 0.1429 & 0.5714 & 0.5000 & 0.5000 & 0.5714 \\ 0.1429 & 0.5714 & 0.2143 & 0.3571 & 0.0714 & 0.2857 & 0.0714 & 0.6429 & 0.4286 & 0.2857 & 0.4286 & 0.5714 & 0.5000 & 0.5714 & 0.1429 & 0.3571 & 0.0714 & 0.5000 & X & 0.0714 & 0.5714 & 0.7143 & 0.1429 & 0.2143 & 0.4286 & 0.1429 & 0.2857 & 0.4286 & 0.4286 & 0.5714 \\ 0.2857 & 0.6429 & 0.4286 & 0.7857 & 0.5000 & 0.4286 & 0.3571 & 0.8571 & 0.8571 & 0.8571 & 0.7857 & 0.9286 & 0.6429 & 0.7857 & 0.7857 & 0.6429 & 0.7857 & 0.8571 & 0.9286 & X & 0.7143 & 0.8571 & 0.8571 & 0.6429 & 0.7143 & 0.1429 & 0.6429 & 0.4286 & 0.7143 & 0.7143 \\ 0.1429 & 0.6429 & 0.2143 & 0.5000 & 0.2143 & 0.2143 & 0.3571 & 0.5000 & 0.4286 & 0.4286 & 0.3571 & 0.5714 & 0.2857 & 0.6429 & 0.2143 & 0.2857 & 0.2857 & 0.4286 & 0.5000 & 0.2857 & X & 0.5714 & 0.3571 & 0.2857 & 0.2857 & 0.2857 & 0.2857 & 0.2857 & 0.4286 & 0.4286 \\ 0.0714 & 0.5714 & 0.2857 & 0.2857 & 0.1429 & 0.2857 & 0.1429 & 0.6429 & 0.3571 & 0.3571 & 0.2857 & 0.5714 & 0.5000 & 0.6429 & 0.1429 & 0.5000 & 0.2143 & 0.2857 & 0.6429 & 0.1429 & 0.6429 & X & 0.1429 & 0.2143 & 0.4286 & 0.1429 & 0.2857 & 0.4286 & 0.3571 & 0.5714 \\ 0.1429 & 0.6429 & 0.4286 & 0.7143 & 0.2143 & 0.3571 & 0.4286 & 0.8571 & 0.7143 & 0.6429 & 0.7143 & 0.9286 & 0.6429 & 0.7857 & 0.5714 & 0.5714 & 0.4286 & 0.6429 & 0.8571 & 0.3571 & 0.7143 & 0.8571 & X & 0.5714 & 0.7143 & 0.1429 & 0.5714 & 0.4286 & 0.7143 & 0.7143 \\ 0.3571 & 0.7857 & 0.4286 & 0.7857 & 0.2143 & 0.4286 & 0.3571 & 0.8571 & 0.9286 & 0.5714 & 0.8571 & 0.9286 & 0.6429 & 0.9286 & 0.5000 & 0.5000 & 0.4286 & 0.7143 & 0.9286 & 0.3571 & 0.8571 & 0.7857 & 0.5714 & X & 0.7857 & 0.3571 & 0.7143 & 0.5714 & 0.7857 & 0.8571 \\ 0.2143 & 0.7143 & 0.5000 & 0.6429 & 0.1429 & 0.3571 & 0.2143 & 0.6429 & 0.5714 & 0.4286 & 0.5714 & 0.7143 & 0.5714 & 0.7143 & 0.4286 & 0.5000 & 0.2857 & 0.5000 & 0.6429 & 0.2857 & 0.7143 & 0.7857 & 0.3571 & 0.3571 & X & 0.3571 & 0.5000 & 0.5714 & 0.5714 & 0.7857 \\ 0.7143 & 0.7857 & 0.5000 & 0.8571 & 0.6429 & 0.6429 & 0.7857 & 0.7857 & 0.8571 & 0.8571 & 0.8571 & 0.8571 & 0.7143 & 0.7143 & 0.7857 & 0.6429 & 0.8571 & 0.8571 & 0.8571 & 0.8571 & 0.7857 & 0.8571 & 0.8571 & 0.7143 & 0.6429 & X & 0.8571 & 0.7143 & 0.7857 & 0.7857 \\ 0.2857 & 0.7143 & 0.2857 & 0.6429 & 0.2857 & 0.4286 & 0.3571 & 0.7143 & 0.7143 & 0.5714 & 0.7143 & 0.7857 & 0.5000 & 0.7857 & 0.3571 & 0.5000 & 0.4286 & 0.5000 & 0.7857 & 0.3571 & 0.7143 & 0.7143 & 0.5714 & 0.6429 & 0.5714 & 0.3571 & X & 0.5714 & 0.7143 & 0.7857 \\ 0.3571 & 0.7857 & 0.3571 & 0.5000 & 0.5000 & 0.5000 & 0.4286 & 0.8571 & 0.6429 & 0.5714 & 0.5714 & 0.8571 & 0.4286 & 0.8571 & 0.5000 & 0.6429 & 0.5714 & 0.5714 & 0.6429 & 0.5714 & 0.8571 & 0.7143 & 0.5714 & 0.5000 & 0.5000 & 0.5000 & 0.7143 & X & 0.6429 & 0.8571 \\ 0.1429 & 0.7143 & 0.2143 & 0.5000 & 0.2857 & 0.2143 & 0.3571 & 0.7143 & 0.5000 & 0.4286 & 0.5000 & 0.7857 & 0.5000 & 0.7143 & 0.5000 & 0.5000 & 0.4286 & 0.5714 & 0.7143 & 0.4286 & 0.7857 & 0.7857 & 0.4286 & 0.2857 & 0.6429 & 0.2857 & 0.2857 & 0.4286 & X & 0.7143 \\ 0.1429 & 0.6429 & 0.2143 & 0.5000 & 0.2143 & 0.2143 & 0.2857 & 0.5714 & 0.5000 & 0.4286 & 0.4286 & 0.7143 & 0.2857 & 0.6429 & 0.2857 & 0.4286 & 0.3571 & 0.4286 & 0.5000 & 0.2857 & 0.7857 & 0.5714 & 0.2857 & 0.2143 & 0.3571 & 0.3571 & 0.4286 & 0.5714 & 0.3571 & X \end{pmatrix} \end{array}$$

Boolean preferences matrix

$$\mathcal{O}_B =$$

	Celtics	*Nets*	*Knicks*	*Sixers*	*Raptors*	*Bulls*	*Cavs*	*Pistons*	*Pacers*	*Bucks*	*Hawks*	*Hornets*	*Heat*	*Magic*	*Wizards*	*Mavs*	*Rockets*	*Grizzlies*	*Pelicans*	*Spurs*	*Nuggets*	*Wolves*	*Thunder*	*Blazers*	*Jazz*	*Warriors*	*Clippers*	*Lakers*	*Suns*	*Kings*
Celtics	X	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
Nets	0	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Knicks	0	1	X	1	0	0	0	1	1	1	1	1	1	1	0	1	0	1	1	0	1	1	0	1	1	0	1	1	1	1
Sixers	0	0	0	X	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
Raptors	0	1	0	1	X	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1
Bulls	0	1	0	1	0	X	0	1	1	0	1	1	1	1	0	1	0	1	1	0	1	1	1	1	1	0	1	1	1	1
Cavs	0	0	0	1	0	0	X	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	1	1	0	1	0	1	1
Pistons	0	0	0	0	0	0	0	X	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Pacers	0	0	0	0	0	0	0	1	X	0	1	1	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0
Bucks	0	0	0	0	0	0	0	1	0	X	1	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
Hawks	0	0	0	0	0	0	0	0	0	0	X	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
Hornets	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Heat	0	1	0	0	0	0	0	1	0	0	0	1	X	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0	1	1
Magic	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Wizards	0	1	0	1	0	0	0	1	1	1	1	1	0	1	X	0	0	1	1	0	1	1	0	0	1	0	0	0	0	1
Mavs	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	X	0	0	1	0	1	1	0	0	0	0	0	0	0	1
Rockets	0	1	0	1	0	0	0	1	1	1	1	1	0	1	1	0	X	1	1	0	1	1	1	0	1	0	0	0	1	0
Grizzlies	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	X	1	0	0	1	0	0	0	0	0	0	0	0
Pelicans	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	1	0	0	0	0	0	0	0	0
Spurs	0	0	0	1	0	0	0	1	1	1	1	1	0	1	1	0	1	1	1	X	1	1	1	0	1	0	0	0	1	1
Nuggets	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0
Wolves	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0
Thunder	0	0	0	1	0	0	0	1	1	0	1	1	0	1	0	0	0	0	1	0	1	1	X	0	1	0	0	0	1	1
Blazers	0	1	0	1	0	0	0	1	1	0	1	1	0	1	0	0	0	1	1	0	1	1	0	X	1	0	1	0	1	1
Jazz	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	X	0	0	0	0	1
Warriors	1	1	0	1	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	X	1	1	1	1
Clippers	0	1	0	0	0	0	0	1	1	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	X	0	1	1
Lakers	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	X	0	1
Suns	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	0	0	X	1
Kings	0	0	0	0	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	X

Draft workouts

Pichat algorithm calculation

The result of the Pichat algorithm has been calculated as follows:

$$
\begin{aligned}
P &= (1+2,3,4,5,12)(2+3,4,6,7,8,10,11)(3+4,5,6,8,9,10,11,12)(4+6,8,9,10,11)(5+7,8)(6+7,8,11)(7+8,9,10,11)(8+9,10,11,12)(10+11)(11+12)\\
&= (1,2,3+1,2,4,5,6,8,9,10,11,12+1,3,4,6,7,8,10,11+2,3,4,5,12+2,3,4,6,7,8,10,11)(4+6,8,9,10,11)(5+7,8)(6+7,8,11)(7+8,9,10,11)(8+9,10,11,12)(10+11)(11+12)\\
&= (1,2,3,4,5,6,7+1,2,3,4,6,7,8+1,2,3,4,7,8,11+1,2,3,5,6,8,9,10,11+1,2,3,6,7,8,9,10,11+1,2,4,5,6,8,9,10,11,12+1,3,4,6,7,8,10,11+2,3,4,5,6,7,12+2,3,4,5,6,8,9,10,11,12+\\
&\quad +2,3,4,5,7,8,11,12+2,3,4,6,7,8,10,11)(8+9,10,11,12)(10+11)(11+12)\\
&= 1,2,3,4,6,7,8,10,12+1,2,3,4,7,8,11+1,2,3,5,6,8,9,10,11+1,2,3,6,7,8,9,10,11+1,2,4,5,6,8,9,10,11,12+1,3,4,6,8,10,11+2,3,4,5,6,7,8,10,12+2,3,4,5,6,7,9,10,11,12+\\
&\quad +2,3,4,5,6,8,9,10,11,12+2,3,4,5,7,8,11,12+2,3,4,6,7,8,10,11
\end{aligned}
$$

Hamming distances

The Hamming distances matrix, calculated with the programmed algorithm in the section called Hamming distances and similarities matrices in this Appendix, is as follows:

	Fultz	*Ball*	*Jackson*	*Fox*	*Tatum*	*Isaac*	*Smith*	*Monk*	*Ntilikina*	*Collins*	*Markannen*	*Allen*
Fultz	0.00	0.15	0.13	0.13	0.13	0.10	0.12	0.12	0.10	0.08	0.12	0.17
Ball	0.15	0.00	0.18	0.15	0.12	0.15	0.17	0.23	0.12	0.13	0.17	0.12
Jackson	0.13	0.18	0.00	0.13	0.20	0.20	0.12	0.22	0.20	0.18	0.25	0.20
Fox	0.13	0.15	0.13	0.00	0.10	0.13	0.12	0.18	0.13	0.15	0.22	0.10
Tatum	0.13	0.12	0.20	0.10	0.00	0.10	0.15	0.18	0.07	0.08	0.12	0.07
Isaac	0.10	0.15	0.20	0.13	0.10	0.00	0.18	0.15	0.10	0.12	0.15	0.10
Smith	0.12	0.17	0.12	0.12	0.15	0.18	0.00	0.17	0.18	0.17	0.20	0.12
Monk	0.12	0.23	0.22	0.18	0.18	0.15	0.17	0.00	0.15	0.20	0.13	0.15
Ntilikina	0.10	0.12	0.20	0.13	0.07	0.10	0.18	0.15	0.00	0.05	0.08	0.10
Collins	0.08	0.13	0.18	0.15	0.08	0.12	0.17	0.20	0.05	0.00	0.13	0.12
Markannen	0.12	0.17	0.25	0.22	0.12	0.15	0.20	0.13	0.08	0.13	0.00	0.15
Allen	0.17	0.12	0.20	0.10	0.07	0.10	0.12	0.15	0.10	0.12	0.15	0.00

Recall that the similarities matrix is built from the Hamming distances matrix as follows:

$$M_S = M_1 - M_H,$$

where M_S is the similarities matrix, M_1 is the matrix with ones in all boxes, and M_H the Hamming distances matrix.

Hungarian algorithm application

The Hungarian algorithm begins with the following square matrix:

	c_1	c_2	c_3	c_4	c_5
g_1	0.9394	0.9394	0.9091	1	1
g_2	0.8485	0.9394	0.9091	0.9394	0.8182
g_3	0.6061	0.6667	0.7273	0.7576	0.6667
g_4	0.6061	0.6364	0.6667	0.6970	0.5758
g_F	1	1	1	1	1

As a fictitious row has been added, the first step is to subtract from each element in the matrix the minimum value in the column:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_F \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 0.3333 & 0.3030 & 0.2424 & 0.3030 & 0.4242 \\ 0.8485 & 0.9394 & 0.9091 & 0.9394 & 0.8182 \\ 0.6061 & 0.6667 & 0.7273 & 0.7576 & 0.6667 \\ 0.6061 & 0.6364 & 0.6667 & 0.6970 & 0.5758 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

The following step is to subtract to each element the minimum value in the row:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_F \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 0.0909 & 0.0606 & 0 & 0.0606 & 0.1818 \\ 0 & 0.0606 & 0 & 0 & 0 \\ 0 & 0.6667 & 0.7273 & 0.7576 & 0.6667 \\ 0 & 0 & 0 & 0 & 0 \\ 0.0909 & 0.0606 & 0.0303 & 0 & 0.1212 \end{pmatrix} \end{array}$$

If an assignation between groups and coaches is possible, then the process is finished. If not, the process has to be iterated until a solution is found.
In bold type below, a possible assignation can be seen:

$$\begin{array}{c} \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_F \end{array} \begin{array}{c} \begin{array}{ccccc} c_1 & c_2 & c_3 & c_4 & c_5 \end{array} \\ \begin{pmatrix} 0.0909 & 0.0606 & \mathbf{0} & 0.0606 & 0.1818 \\ 0 & 0.0606 & 0 & 0 & \mathbf{0} \\ \mathbf{0} & 0.6667 & 0.7273 & 0.7576 & 0.6667 \\ 0 & \mathbf{0} & 0 & 0 & 0 \\ 0.0909 & 0.0606 & 0.0303 & \mathbf{0} & 0.1212 \end{pmatrix} \end{array}$$

Therefore, the algorithm is stopped because an optimal assignation has been found.

Programming codes for the algorithms

Creation of the boolean order matrix

Notation:

- M is the initial fuzzy matrix.

- B is the final boolean matrix.

- orderMatrix is the final fuzzy matrix.

- c1,...,c14 are the matrices for each criteria.

```
#include<stdlib.h>
#include<stdio.h>
#include<math.h>

void fillMatrix(double**,int,int);
double** createMemory(int,int);
int** createIntMemory(int,int);
void criteriaMatrix(double**,double**,int,int);
void finalMatrix(double**,double**,double**,double**,double**,
     double**,double**,double**,double**,double**,
     double**,double**,double**,double**,double**,int);
void printMatrix(double**,int,int);
void printIntMatrix(int**,int,int);
void alphaCut(int**,double**,int,int,double);


int main(void){
  double **M,**c1,**c2,**c3,**c4,**c5,**c6,**c7,**c8,
         **c9,**c10,**c11,**c12,**c13,**c14,**orderMatrix;
  int i,j,dim;
  int **B;
  double alpha;
  FILE *f;

  f = fopen("printMalgrange.txt","w");

  alpha = 0.65;

  /*Create space for the matrices*/
  M = createMemory(30,14);
  c1 = createMemory(30,30);
  c2 = createMemory(30,30);
  c3 = createMemory(30,30);
```

```
34   c4 = createMemory(30,30);
35   c5 = createMemory(30,30);
36   c6 = createMemory(30,30);
37   c7 = createMemory(30,30);
38   c8 = createMemory(30,30);
39   c9 = createMemory(30,30);
40   c10 = createMemory(30,30);
41   c11 = createMemory(30,30);
42   c12 = createMemory(30,30);
43   c13 = createMemory(30,30);
44   c14 = createMemory(30,30);
45   orderMatrix = createMemory(30,30);
46   B = createIntMemory(30,30);
47
48
49   /*Fill in the initial matrix*/
50   fillMatrix(M,30,14);
51
52   /*Fill in criteria matrices*/
53   criteriaMatrix(c1,M,30,0);
54   criteriaMatrix(c2,M,30,1);
55   criteriaMatrix(c3,M,30,2);
56   criteriaMatrix(c4,M,30,3);
57   criteriaMatrix(c5,M,30,4);
58   criteriaMatrix(c6,M,0,5);
59   criteriaMatrix(c7,M,30,6);
60   criteriaMatrix(c8,M,30,7);
61   criteriaMatrix(c9,M,30,8);
62   criteriaMatrix(c10,M,30,9);
63   criteriaMatrix(c11,M,30,10);
64   criteriaMatrix(c12,M,30,11);
65   criteriaMatrix(c13,M,30,12);
66   criteriaMatrix(c14,M,30,13);
67
68   /*Build the final matrix*/
69   finalMatrix(orderMatrix,c1,c2,c3,c4,c5,c6,c7,c8,c9,
70           c10,c11,c12,c13,c14,30);
71
72   /*Print the final matrix*/
73   fprintf(f,"Final matrix:\n");
74   for(i=0;i<30;i++){
75     for(j=0;j<30;j++){
76       fprintf(f,"%6.4lf    ",orderMatrix[i][j]);
77     }
78     fprintf(f,"\n");
```

```
79    }
80    /*printMatrix(orderMatrix,30,30,);*/
81
82    /*Apply the alpha-cut to our final matrix*/
83    alphaCut(B,orderMatrix,30,30,alpha);
84
85    /*Print the boolean matrix*/
86    fprintf(f,"\n\n\nBoolean matrix:\n");
87    for(i=0;i<30;i++){
88      for(j=0;j<30;j++){
89        fprintf(f,"%d   ",B[i][j]);
90      }
91      fprintf(f,"\n");
92    }
93
94    fclose(f);
95
96    return 0;
97 }
98
99
100 /*-------------------------------------------------------------------------*/
101
102 double** createMemory(int rows,int cols){
103    int i;
104    double **M;
105
106   M = (double**)calloc(rows, sizeof(double*));
107    for(i=0;i<rows;i++)
108      M[i] = (double*)calloc(cols, sizeof(double));
109
110    return M;
111 }
112
113
114 /*-------------------------------------------------------------------------*/
115
116 int** createIntMemory(int rows,int cols){
117    int i;
118    int **M;
119
120   M = (int**)calloc(rows, sizeof(int*));
121    for(i=0;i<rows;i++)
122      M[i] = (int*)calloc(cols, sizeof(int));
123
```

```
124   return M;
125 }
126
127
128 /*-------------------------------------------------------------------*/
129
130 void fillMatrix(double** M,int nrows,int ncols){
131   int i,j;
132   double value;
133   FILE *f;
134
135   /*Read the matrix in the file OrderInitialMatrix.txt*/
136   f = fopen("OrderInitialMatrix.txt","r");
137
138   i=0;
139   j=0;
140
141   value = getc(f);
142   while(value!=EOF){
143     M[i][j]=value;
144
145     j++;
146     value = getc(f);
147
148     if(j==30){
149       i++;
150       j=0;
151     }
152   }
153
154   fclose(f);
155
156   return ;
157 }
158
159
160 /*-------------------------------------------------------------------*/
161
162
163 void criteriaMatrix(double** c, double** M, int nrows, int col){
164   int i,j;
165
166   for(i=0;i<nrows;i++){
167     for(j=0;j<nrows;j++){
168       if(M[i][col] >= M[j][col])
```

```
169           c[i][j] = 1;
170         else
171           c[i][j]=0;
172       }
173     }
174
175     return ;
176 }
177
178
179 /*--------------------------------------------------------------------*/
180
181
182 void finalMatrix(double **orderMatrix,double **c1,double **c2,
183       double **c3,double **c4,double **c5,double **c6,
184       double **c7,double **c8,double **c9,double **c10,
185       double **c11,double **c12,double **c13,double **c14, int n){
186     int i,j;
187
188     for(i=0;i<n;i++){
189       for(j=0;j<n;j++){
190         orderMatrix[i][j] = (c1[i][j]+c2[i][j]+c3[i][j]+c4[i][j]+
191                   c5[i][j]+c6[i][j]+c7[i][j]+c8[i][j]+
192                   c9[i][j]+c10[i][j]+c11[i][j]+c12[i][j]+
193                   c13[i][j]+c14[i][j])/14.;
194       }
195     }
196
197     return ;
198 }
199
200
201 /*--------------------------------------------------------------------*/
202
203
204 void printMatrix(double **M,int nrows,int ncols){
205     int i,j;
206
207     for(i=0;i<nrows;i++){
208       for(j=0;j<ncols;j++){
209         printf("%6.4lf    ",M[i][j]);
210       }
211       printf("\n");
212     }
213
```

```
214   return ;
215 }
216
217
218 /*-------------------------------------------------------------------*/
219
220
221 void printIntMatrix(int **M,int nrows,int ncols){
222   int i,j;
223
224   for(i=0;i<nrows;i++){
225     for(j=0;j<ncols;j++){
226       printf("%d    ",M[i][j]);
227     }
228     printf("\n");
229   }
230
231   return ;
232 }
233
234
235 /*-------------------------------------------------------------------*/
236
237
238 void alphaCut(int **B,double **M,int nrows,int ncols,double alpha){
239   int i,j;
240
241   for(i=0;i<nrows;i++){
242     for(j=0;j<ncols;j++){
243       if(M[i][j] >= alpha)
244         B[i][j] = 1;
245       else
246         B[i][j] = 0;
247     }
248   }
249
250   return ;
251 }
```

Hamming distances and similarities matrices

Notation:

- Mqualities is the matrix containing the qualities for each player.
- Mone is a matrix with all elements equal to 1.
- MHamming is the Hamming distances matrix.
- MSimil is the similarities matrix.

```
1 #include<stdlib.h>
2 #include<stdio.h>
3 #include<math.h>
4 #include<float.h>
5
6 double** createMemory(int,int);
7 void declareMqualities(double**);
8 void declareMone(double**);
9 void HammingDist(double**,double**,int,int);
10 void substractMatrix(double**,double**,double**,int,int);
11 void printMatrix(double**,int,int);
12 void freeUpMemory(double**,int);
13
14
15 int main(void){
16   double **Mqualities, **MHamming, **Mone, **MSimil;
17   int players,qualities,i,j;
18   FILE *f;
19
20   f = fopen("HammingSimilMatrix.txt","w");
21
22   /*Read the number of elements in set X*/
23   players = 12;
24   /*Read the number of elements in set Y*/
25   qualities = 12;
26
27   /*Create memory for all the matrices involved in the algorithm*/
28   Mqualities = createMemory(players,qualities);
29   MHamming = createMemory(players,players);
30   Mone = createMemory(players,players);
31   MSimil = createMemory(players,players);
32
33   /*Declaration of the matrices*/
34   declareMqualities(Mqualities);
35   declareMone(Mone);
```

```
36
37    /*Calculation of Hamming matrix*/
38    HammingDist(MHamming, Mqualities, players, qualities);
39
40    /*Calculation of Similarities Matrix*/
41    substractMatrix(MSimil, Mone, MHamming, players, players);
42
43    /*Print the Hamming Matrix*/
44    printMatrix(MHamming, players, players);
45
46    /*Free up memory for all matrices involved in the algorithm*/
47    freeUpMemory(Mqualities, players);
48    freeUpMemory(MHamming, players);
49    freeUpMemory(Mone, players);
50    freeUpMemory(MSimil, players);
51
52    return 0;
53 }
54
55
56
57
58 /*-------------------------------------------------------------------*/
59
60 double** createMemory(int rows, int cols){
61    int i;
62    double **M;
63
64   M = (double**)calloc(rows, sizeof(double*));
65    for(i=0;i<rows;i++)
66      M[i] = (double*)calloc(cols, sizeof(double));
67
68    return M;
69 }
70
71
72 /*-------------------------------------------------------------------*/
73
74 void declareMqualities(double** Mqualities){
75    int i,j;
76    double value;
77    FILE *f;
78
79    /*Read the matrix in the file HammingQualities.txt*/
80    f = fopen("HammingQualities.txt","r");
```

```
 81
 82    i=0;
 83    j=0;
 84
 85    value = getc(f);
 86    while(value!=EOF){
 87      Mqualities[i][j]=value;
 88
 89      j++;
 90      value = getc(f);
 91
 92      if(j==12){
 93        i++;
 94        j=0;
 95      }
 96    }
 97
 98    fclose(f);
 99
100    return ;
101 }
102
103
104 /*-------------------------------------------------------------------------*/
105
106 void declareMone(double** Mone){
107    int i,j;
108
109    for(i=0;i<12;i++){
110      for(j=0;j<12;j++){
111        Mone[i][j]=1;
112      }
113    }
114
115    return;
116 }
117
118
119 /*-------------------------------------------------------------------------*/
120
121
122 void HammingDist(double** Hamming, double** origin,
123        int dimHamming,int ncriteria){
124    int i,j,k;
125    double dist;
```

```
126
127   for(i=0;i<dimHamming;i++){
128     for(j=0;j<dimHamming;j++){
129       for(k=0;k<ncriteria;k++){
130         dist = origin[i][k]-origin[j][k];
131         if(dist<0)
132           dist = -dist;
133         Hamming[i][j] = Hamming[i][j] + dist;
134       }
135       Hamming[i][j] = (Hamming[i][j])/12;
136     }
137   }
138
139   return ;
140 }
141
142
143 /*-------------------------------------------------------------------*/
144
145 void substractMatrix(double** substr,double** M1,double** M2,
146          int rows,int cols){
147   int i,j;
148
149   for(i=0;i<rows;i++){
150     for(j=0;j<cols;j++){
151       substr[i][j] = M1[i][j] - M2[i][j];
152     }
153   }
154
155   return ;
156 }
157
158
159 /*-------------------------------------------------------------------*/
160
161 void printMatrix(double** M,int rows,int cols){
162   int i,j;
163   FILE *f;
164
165   f = fopen("HammingSimilMatrix.txt","w");
166
167   for(i=0;i<rows;i++){
168     for(j=0;j<cols;j++)
169       fprintf(f,"%4.21f    ",M[i][j]);
170     fprintf(f,"\n");
```

```
171     }
172
173     fclose(f);
174
175     return ;
176 }
177
178
179 /*-------------------------------------------------------------------------*/
180
181 void freeUpMemory(double** M, int rows){
182     int i;
183
184     for(i=0;i<rows;i++)
185         free(M[i]);
186     free(M);
187
188     return ;
189 }
```

Forgotten Effects algorithm

Notation:

- Set X is the set of causes.

- Mx is the relation matrix between causes and causes.

- Set Y is the set of effects.

- My is the relation matrix between effects and effects.

- Rxy is the relation matrix between causes and effects.

```
#include<stdlib.h>
#include<stdio.h>
#include<math.h>
#include<float.h>

double** createMemory(int,int);
void declareMx(double**);
void declareMy(double**);
void declareRxy(double**);
void MaxMinConvolution(double**,double**,double**,int,int,int);
void substractMatrix(double**,double**,double**,int,int);
void printMatrix(double**,int,int);
void freeUpMemory(double**,int);
double min(double,double);


int main(void){
  double **Mx, **My, **Rxy, **MxConvRxy,
         **CumulativeEffects, **ForgottenEffects;
  int elementsSetX,elementsSetY,i,j;
  FILE *f;

  f = fopen("ForgottenPrint.txt","w");

  /*read the number of elements in set X*/
  elementsSetX = 11;
  /*read the number of elements in set Y*/
  elementsSetY = 5;

  /*Create memory for all the matrices involved in the algorithm*/
  Mx = createMemory(elementsSetX,elementsSetX);
  My = createMemory(elementsSetY,elementsSetY);
  Rxy = createMemory(elementsSetX,elementsSetY);
```

```
34    MxConvRxy = createMemory(elementsSetX,elementsSetY);
35    CumulativeEffects = createMemory(elementsSetX,elementsSetY);
36    ForgottenEffects = createMemory(elementsSetX,elementsSetY);
37
38    /*Declaration of the matrices*/
39    declareMx(Mx);
40    declareMy(My);
41    declareRxy(Rxy);
42
43    /*MaxMinConv for Mx and Rxy; output=MxConvRxy*/
44    MaxMinConvolution(MxConvRxy,Mx,Rxy,elementsSetX,
45          elementsSetX,elementsSetY);
46
47    /*MaxMinConv for MxConvRxy and My; output=CumulativeEffects*/
48    MaxMinConvolution(CumulativeEffects,MxConvRxy,My,elementsSetX,
49          elementsSetY,elementsSetY);
50
51    /*subtractMatrix for CumulativeEffects and Rxy; output=ForgottenEffects*/
52    substractMatrix(ForgottenEffects,CumulativeEffects,Rxy,
53         elementsSetX,elementsSetY);
54
55    /*Print Forgotten Effects matrix*/
56    printMatrix(ForgottenEffects,elementsSetX,elementsSetY);
57
58    /*Free up memory for all matrices involved in the algorithm*/
59    freeUpMemory(Mx,elementsSetX);
60    freeUpMemory(My,elementsSetY);
61    freeUpMemory(Rxy,elementsSetX);
62    freeUpMemory(MxConvRxy,elementsSetX);
63    freeUpMemory(CumulativeEffects,elementsSetX);
64    freeUpMemory(ForgottenEffects,elementsSetX);
65
66    return 0;
67 }
68
69
70
71
72 /*-------------------------------------------------------------------*/
73
74 double** createMemory(int rows,int cols){
75    int i;
76    double **M;
77
78    M = (double**)calloc(rows, sizeof(double*));
```

```
79    for(i=0;i<rows;i++)
80      M[i] = (double*)calloc(cols, sizeof(double));
81
82    return M;
83  }
84
85
86  /*-------------------------------------------------------------------*/
87
88  void declareMx(double** Mx){
89    int i,j;
90    double value;
91    FILE *f;
92
93    /*Read the matrix in the file ForgottenMx.txt*/
94    f = fopen("ForgottenMx.txt","r");
95
96    i=0;
97    j=0;
98
99    value = getc(f);
100   while(value!=EOF){
101     Mx[i][j]=value;
102
103     j++;
104     value = getc(f);
105
106     if(j==11){
107       i++;
108       j=0;
109     }
110   }
111
112   fclose(f);
113
114   return ;
115 }
116
117
118 /*-------------------------------------------------------------------*/
119
120 void declareMy(double** My){
121   int i,j;
122   double value;
123   FILE *f;
```

```
124
125   /*Read the matrix in the file ForgottenMy.txt*/
126   f = fopen("ForgottenMy.txt","r");
127
128   i=0;
129   j=0;
130
131   value = getc(f);
132   while(value!=EOF){
133     My[i][j]=value;
134
135     j++;
136     value = getc(f);
137
138     if(j==5){
139       i++;
140       j=0;
141     }
142   }
143
144   fclose(f);
145
146   return ;
147 }
148
149
150 /*-------------------------------------------------------------------------*/
151
152 void declareRxy(double** Rxy){
153   int i,j;
154   double value;
155   FILE *f;
156
157   /*Read the matrix in the file ForgottenRxy.txt*/
158   f = fopen("ForgottenRxy.txt","r");
159
160   i=0;
161   j=0;
162
163   value = getc(f);
164   while(value!=EOF){
165     Rxy[i][j]=value;
166
167     j++;
168     value = getc(f);
```

```
169
170       if(j==5){
171         i++;
172         j=0;
173       }
174     }
175
176     fclose(f);
177
178     return ;
179 }
180
181
182 /*-------------------------------------------------------------------------*/
183
184 void MaxMinConvolution(double** conv,double** M1,double** M2,
185         int rowsM1,int colsM1,int colsM2){
186     int i,j,k;
187     double m;
188
189
190     for(i=0;i<rowsM1;i++){
191       for(j=0;j<colsM2;j++){
192         conv[i][j] = 0;
193         for(k=0;k<colsM1;k++){
194           m = min(M1[i][k],M2[k][j]);
195           if(m>conv[i][j]){
196             conv[i][j] = m;
197           }
198         }
199       }
200     }
201
202     return ;
203 }
204
205
206 /*-------------------------------------------------------------------------*/
207
208 void substractMatrix(double** substr,double** M1,double** M2,
209         int rows,int cols){
210     int i,j;
211
212     for(i=0;i<rows;i++)
213       for(j=0;j<cols;j++)
```

```
214         substr[i][j] = M1[i][j] - M2[i][j];
215
216     return ;
217 }
218
219
220 /*-------------------------------------------------------------------------*/
221
222 void printMatrix(double** M,int rows,int cols){
223     int i,j;
224     FILE *f;
225
226     f = fopen("ForgottenPrint.txt","w");
227
228     for(i=0;i<rows;i++){
229       for(j=0;j<cols;j++)
230         fprintf(f,"%4.21f    ", M[i][j]);
231       fprintf(f,"\n");
232     }
233
234     fclose(f);
235
236     return ;
237 }
238
239
240 /*-------------------------------------------------------------------------*/
241
242 void freeUpMemory(double** M,int rows){
243     int i;
244
245     for(i=0;i<rows;i++)
246       free(M[i]);
247     free(M);
248
249     return ;
250 }
251
252
253 /*-------------------------------------------------------------------------*/
254
255 double min(double a,double b){
256     double r;
257
258     if(a<=b)
```

```
259       r=a;
260    if(a>b)
261       r=b;
262
263    return r;
264 }
```