Universidad del País Vasco
Euskal Herriko Unibertsitatea

ZTF-FCT
Zientzia eta Teknologia Fakultatea
Facultad de Ciencia y Tecnología

The assignment problem for Kidney Paired Donation

Final Degree Dissertation
Degree in Mathematics

Miren Lur Barquin Torre

Supervisor:
María Merino Maestre

Leioa, 19 February 2020

Contents

Acknowledgements

This dissertation would not have been possible without the guidance and help of my supervisor María Merino Maestre; I would like to thank her for the time and dedication.

I would like to express my appreciation to Jordi Massó Carreras for his stimulating informative article, which initially motivated this dissertation.

I would like to acknowledge the Spanish National Transplant Organization (ONT) for their kind attention, and the provided essential medical information and help. I would like to thank also Elixabete Arrese Arratibel and Ester Suñen Pardo, from the Immunology, Microbiology, and Parasitology Department of the University of the Basque Country, for their responses to the difficulties faced on immunology and HLA.

I would like to express my sincere gratitude to the University of the Basque Country and the University of Primorska for the opportunity of being a student in their institutions, and for providing me the necessary academic training to commence my career as mathematician, and grow as a student and as a person.

Lastly, I am also grateful to my parents, brother and close friends, for respecting always my personal space in these demanding years.

Abstract

For patients with end-stage renal disease, kidney transplant is the best available treatment, but due to graft accessibility and compatibility limitations, fewer transplants than desired are performed. As a solution, F. T. Rapaport proposed in 1986 the idea of a paired kidney exchange with living donors. In this dissertation we present the adaptation of the Top Trading Cycles algorithm of Gale, published in 1974 by L. Shapley and H. Scarf for an economic model of trading, to the paired kidney exchange idea of Rapaport. Such adaptation is due to A. E. Roth, T. Sönmez and M. U. Ünver (2004). Gale's TTC algorithm allows to achieve an assignation of the kidneys among the patients that ensures the proper utilization of the grafts, the satisfaction of the patients, and the non-manipulability of the mechanism. In this paper we deeply study the main theoretical properties of the algorithm. Later, we implement a C++ program for the mentioned algorithm, and perform 140 simulations, varying the dimensions of the problem and the preference input matrices generated according to real data from the Spanish National Transplant Organization. Finally, we have analysed the results and draw some conclusions and further research.

Resumen

El mejor tratamiento disponible para pacientes con enfermedad renal en etapa terminal es el trasplante de riñón, pero debido a las limitaciones en la compatibilidad y accesibilidad de los órganos, se llevan a cabo menos trasplantes de los deseados. Como solución, F.T. Rapaport propuso en 1986 el trasplante cruzado de riñones de donantes vivos. En este trabajo presentamos la adaptación del algoritmo 'Top Trading Cycles' de Gale, publicado en 1974 por L. Shapley y H. Scarf para un modelo económico de intercambio, a la idea de trasplantes cruzados de riñones de Rapaport. Dicha adaptación se debe a A. E. Roth, T. Sönmez y M. U. Ünver (2004). El algoritmo TTC de Gale permite lograr una asignación de los riñones entre los pacientes que asegura el uso adecuado de los órganos, la satisfacción de los pacientes y la no manipulabilidad del mecanismo. En esta disertación hemos estudiado a fondo las principales propiedades teóricas del algoritmo. Posteriormente, hemos implementado un programa en C++ para el algoritmo anterior y ejecutado 140 simulaciones variando las dimensiones del problema y las matrices de entrada generadas de acuerdo a datos reales de la Organización Nacional de Trasplantes. Finalmente, hemos analizado los resultados y apuntado algunas conclusiones y futura investigación.

Chapter 1

Introduction

1.1 Transplant context

In medicine, a transplant is a complex medical treatment that consists on replacing a sick organ, that risks a person's life, with another one that comes from somebody else and operates properly. By 2019, Spain leads organ donation worldwide for its twenty-seventh consecutive year, as stated by the World Transplant Registry managed by the Spanish National Transplant Organization (ONT), see [22], in collaboration with the World Health Organization (WHO), see [51]. The number of transplants in the world continues raising, but much slower than desired. Despite in the last year the total number of transplants in the world raised by 2.3% regarding the previous one, barely 10% of the necessities of transplantations in the world, estimated by the WHO, are covered. Among all the organ transplants, the kidney transplant stands out globally for its frequency.

For patients with End-Stage Renal Disease (ESRD) (those for which their kidneys carry out less than the 10% of their function), the kidney transplant is the best available treatment. Since even though thanks to the other only treatment for terminal kidney failure, dialysis, they could live many years, their life quality decreases significantly. The average period of waiting to receive a transplant is of 18 months, a long lapse in which some patients lose their life (around 8% in Spain) or their illness worsens, and they stop being potential kidney recipients.

Due to the shortage of organs, fewer transplants are executed than desired. There are two sources from which kidney transplants can come, on the one hand from donors in encephalic death (the most frequent in Spain), and on the other, from living donors. In 2018, in Spain, just an 8.9% of the kidney transplants were from living donors according to the ONT, that is, 6.3 per million of population (PMP) from the total amount of 70.8 kidney transplants PMP, see [26], whereas, worldwide, 36.5% of the organs in kidney transplantations came from living donors. In 2018, the Basque Country ranked third in the Spanish autonomous communities ranking on living donor renal transplants, with 8.7 PMP.

Spain is considerably efficient in obtaining organs from donors in encephalic death, being its donation rate practically the double of the one obtained in most European countries; that is why the limited proportion of living donors amazes. The efficiency on donations in encephalic death may have made the Spanish system not to pay enough attention to the donation from living individuals, which at present brings on a problem resulting from the decrease of traffic accidents. Namely, it implies less overall deceased donors, and especially, an important decrease on the number of young deceased donors; which makes the average age of the donors in encephalic death grow, reducing the life expectancy of the transplanted organ.

The quality and success probability of a kidney transplant are generally higher when the organ comes from a living donor rather than from a deceased one. Furthermore, for recipients aged 60 years or older, cadaveric donor renal transplantation never provides better outcomes than living donor one. In the direct kidney donation between alive, the patient usually receives one of the two kidneys of a family member or a friend. Unfortunately, not every person disposed to become renal donor, and healthy enough to do so, can donate a kidney to his or her intended recipient. To perform the procedure, the kidney must be feasible on medical grounds for the patient; this feasibility depends on the immunological compatibilities between the patient and the donor, specially on blood groups (ABO blood-type) and tissue types (human leukocyte antigen (HLA) type). The HLA antigens' essential role lies on defense against microorganisms; they are genes essential to normal function of the immune response, see [38]. The HLA incompatibility is one of the major causes of organ transplant rejections. In the event of any incompatibility, the transplant is not possible, and the kidney of the donor disappears from the system (the donor returns home) while the patient commonly enters (or remains on) the queue for a cadaver kidney. These incompatibilities constitute approximately one-third of the patients. It is important to point out that buying and selling kidneys is against the law in almost every country of the world (in Spain according to the established by Article 2 of law 30/1979 about graft extraction and transplantation, see [12]).

Until very few years ago, this was the only type of donation performed between alive, but in 1986, the doctor Felix T. Rapaport (1929-2001), was the first one to propose a paired kidney exchange program of living donors in the attempt of making the most of the rejected donors. He was graduate magna cum laude from New York University (NYU) undergraduate school

(1951) and medical school (1954), and won the Medawar Prize in 1998, the highest distinction of the international Transplantation Society, see [41]. The idea is the following: suppose that a patient who needs a kidney transplant obtains incompatibility results in the analysis regarding the kidney that the partner would provide. Shortly after, the same doctor receives another pair of patient-donor also incompatible. The thought of doing a paired donation transplant arises then, possible in the case that the kidney of the first donor is compatible with the second recipient and the first recipient with the second donor. Moreover, longer cycles, involving three or more couples, could be fulfilled. This new method of donation lies on the fact that living donor kidneys can be assigned simultaneously, while cadaver kidneys cannot. The Massachusetts General Hospital held the first kidney paired donation transplant in the 25$^{\text{th}}$ of February 2003 while the first transplant of this category in Spain took place in 2009 in the Clínic Hospital of Barcelona and Virgen de las Nieves Hospital in Granada, see [5].

Lately, several centres in the United States, Europe, and Asia have started to centralise the information about the patient-donor pairs to be able to perform transplants of kidneys from living donors systematically. Up to the moment in which the reference [18] was published, the longest cycle of paired kidney exchange had been held in 2010, in the Northwestern Memorial Hospital of Chicago, involving eight patients and eight donors. In Spain, the longest chain up to the present succeeded in 2013, with 6 transplanted patients. On the performance intervened 5 patient-donor pairs, a 'Good Samaritan' donor (who donates a kidney to a patient who does not know), and a recipient from the cadaver waiting list which closed the cycle, see [24] .

1.2 Literature review

The kidney allocation problem for a paired kidney exchange program is a variant of the 'house allocation problem' proposed by Lloyd Shapley and Herbert Scarf (1974) in the *Journal of Mathematical Economics*, see [39]. Such economic model of the allocation problem with indivisible commodities was later deeply studied also by some important economists including Alvin E. Roth (1982, 1977), Andrew Postlewaite (1977), Jun Wako (1984, 1991), and Jinpeng Ma (1994), see [34, 36, 45, 46, 16]. Each one of them took from Shapley and Scarf [39] as the basis for their papers, and later proceeded focusing mostly on the core defined by weak domination, competitive allocations, and strategy-proofness of the mechanism. In 2004, Alvin E. Roth together with two other economists, Tayfun Sönmez and M. Utku Ünver, adapted for the first time the previous model to the paired kidney exchange problem in [37]. Later in 2010 and 2013, Jordi Massó wrote on this same application of the model in [18] and [19]; being the first one the inspiration article for this dissertation. He was post-doctoral fellow at the University of Pittsburgh with Alvin E. Roth between 1988-1989, and nowadays is professor of Economics at the Universitat Autónoma de Barcelona. Throughout the whole dissertation several areas of mathematics are deliberated such as graph theory, game theory, discrete mathematics, probability, statistics, and mathematical programming.

1.3 Objectives

The general aim of this dissertation is to implement the Top Trading Cycles (TTC) algorithm of Gale to the paired kidney donation. To achieve it, the specific objectives of this paper are: (*i*) to understand the existing problems on kidney donation, (*ii*) to study Gale's TTC algorithm and its theoretical properties, (*iii*) to replicate preference matrices utilising real data in order to perform simulations, to develop the code which automates the algorithm, and to study its performance.

1.4 Organization of the dissertation

The coming dissertation starts by presenting the assignment problem of agents to indivisible objects based on the economic model presented by L. Shapley and H. Scarf (1974) in [39]. This concept and all the basic initial needed definitions and properties will be presented in Chapter 2. The Top Trading Cycles algorithm of Gale, one of the most important notions in this paper, is presented in Chapter 3. First, the main algorithm is introduced to study later the competitive allocations and some properties as the emptiness of the core and manipulability of the algorithm. Then, the previous ideas will lead into the properties the algorithm holds for the benefit of the agents, and proper utilisation of the existing objects. Chapter 4 is dedicated to the computational implementation of the Top Trading Cycles algorithm of Gale to the paired kidney exchange context, and to several simulations and results. Lastly, the conclusions of the dissertation are developed.

There are four appendixes. In Appendix A, some useful proofs for several lemmas from Chapter 3 are given. In Appendix B is described in detail how to create a preference matrix, based on real data, that is the input for the main program, presented in Appendix C, which solves the problem. Finally, in Appendix D the detailed results of the simulations are shown.

Chapter 2

Fundaments on the Assignment Problem

The main theory from which this dissertation emanates was developed by Lloyd S. Shapley and Herbert Scarf (1974) in the *Journal of Mathematical Economics* [39]; an economic model of trading in indivisible commodities from the field of game-theory. Lloyd S. Shapley was awarded the Nobel Prize in Economics 2012 together with the economist Alvin E. Roth 'for their contributions to the theory of stable allocations and the practice of market design', see [42].

This dissertation is focused in the adaptation of the market model to the transplantation problem, in particular, to the paired kidney exchange problem on living donors. Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver were the first ones to propose in 2004 this interpretation, see [37]. Notice that the goods are the kidneys, and they are exchanged without money or payment. In fact, buying and selling organs is a completely illegal activity in almost every country of the world.

Even though this dissertation will be centred mostly on the kidney exchange problem, the concepts can very similarly be applied to other assignment problems with indivisible objects in several daily life fields completely different from medicine. In economics, for the house allocation problem with only existing tenants, see [39], and house allocation model with existing tenants as well as new applicants, see [1], in the work environment for job distribution, see [8], and in education, for the school choice problem, see [2]; among many others.

2.1 The assignment problem of agents to indivisible objects.

Let us consider n traders in a sort of market in which the goods are freely transferable, but a customer never wants more than one item, and owning any item is strictly preferred to owning no items. It is important to consider also that no agent prefers owning several items to owning the most preferred of these items. Each trader owns an indivisible commodity to barter with, and there is no money or other means of exchange. Then the sole purpose of the market activity is to consider the essentially ordinal preferences of the traders among the indivisible objects available, and reallocate their ownership. At least one outcome of this redistribution is such that no subset of traders can improve it by trading their initial objects among themselves.

Let a finite set with n *agents* be denoted by

$$A = \{a_1, ..., a_n\}$$

and a finite set with m $(m \geq n)$ indivisible *objects* by

$$O = \{o_1, ..., o_m\}.$$

Definition 1. An *allocation* α: $A \longrightarrow O$ is a map that assigns to each agent an object such that no object corresponds to more than one agent.

Remark 1. An allocation is an injective map. It is surjective if the sets A and O have the same number of elements.

In this dissertation, the dimension of the sets A and O will be equal such that $|A| = |O| = n$, and each agent in A owns initially exactly one object from O. Therefore, an allocation in the kidney exchange problem is simply a permutation of the kidneys among the patients.

Definition 2. A *graph* G is a pair $G = (N, E)$ consisting of a finite non-empty set N and a set E of two-element subsets of N. The elements of N are called graph nodes or vertices. An element $e = \{a, b\}$ of E is called an edge or arc with end vertices a and b. A graph in which the edges are ordered pairs of nodes is called *directed graph.*

Definition 3. A *cycle* of a graph G, is a sequence of nodes of G that forms a path such that the first node of the path corresponds to the last one.

Definition 4. An *initial assignment* of agents to objects is a map $\mu : A \longrightarrow O$ such that $\mu(a_i) = o_i$ for $i = 1, ..., n$. That is, μ describes all n initial agent-object couples $(a_1, o_1), ..., (a_n, o_n)$ in the assignment problem. It is also called *initial endowment.*

Notice that, an allocation or assignment can be seen as a permutation of the initial endowment μ.

Example 1. *Given four patient-donor couples:*

$$(a_1, o_1),\ (a_2, o_2),\ (a_3, o_3),\ (a_4, o_4),$$

then the agents set and the objects set are respectively:

$$A = \{a_1, a_2, a_3, a_4\} \quad \text{and} \quad O = \{o_1, o_2, o_3, o_4\},$$

and the initial endowment is:

$$\mu = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ o_1 & o_2 & o_3 & o_4 \end{pmatrix},$$

while an assignment example is:

$$\alpha = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ o_3 & o_4 & o_2 & o_1 \end{pmatrix}.$$

In the initial endowment of a problem in kidney transplantation context, assume that for $i = 1, ..., n$, the patient a_i is incompatible with the kidney o_i from donor i; otherwise, the transplant would be carried out between the initial patient-donor couple, and the pair would not be in the assignment problem. Additionally, we will consider that there is no patient entering the problem with more than one incompatible donor.

Remark 2. For kidney o_i being incompatible with patient a_i, $\alpha(a_i) = o_i = \mu(a_i)$ symbolizes that patient i does not obtain any kidney.

Definition 5. Let $a \in A$. The *strict preference*, or simply *preference*, P_a is a transitive, asymmetric, and complete binary relation that represents the strict order or ranking on the set of objects O.

Remark 3. Let $a \in A$.

- The order on the set of objects being strict for each agent in A implies that no agent is indifferent between any two objects.

- P_a is a transitive relation, therefore for all o_i, o_j, and o_k in the set of objects O, $o_i P_a o_j$ and $o_j P_a o_k$ together imply straightforward that $o_i P_a o_k$.

- P_a is an asymmetric relation, hence distinct objects are never both related to one another, i.e. if $oP_a o'$ holds then $o' P_a o$ does not, and vice versa, for $o, o' \in O$ and $o \neq o'$.

- The relation is complete, so very two elements in O are related with each other in some way, i.e. $\forall\ o, o' \in O$ either $oP_a o'$ or $o' P_a o$.

In the transplants context, P_a reflects the degree of desirability for patient a of the kidneys from the donors. The doctor of the sufferer is the one settling the order over the available kidney set, depending on the compatibility of each organ with the patient. Notation:

· $o_j P_{a_i} o_{j'}$ denotes that the kidney from donor j is beforehand better for patient i than the one from donor j', due to different medical factors.

In particular:

· $o_j P_{a_i} o_i$ denotes that kidney o_j and patient a_i are compatible.

· $o_i P_{a_i} o_j$ denotes that kidney o_j and patient a_i are incompatible.

That is because, if the kidney from donor j is worse for patient i than the one from its initial pair i, a_i would be receiving not as good option of kidney as the first alternative o_i.

Example 2.

Suppose that, for the four couples in the previous example, the doctor determines their preferences as Table 1 shows. The preference ranking of each patient is represented by columns in descending order (the most preferable kidney on top), and the kidneys in red cells represent the initial endowment of each patient, previously designated by μ.

Table 1: Preferences.

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}
o_2	o_3	o_1	o_2
o_4	o_2	o_4	o_3
o_1	o_1	o_2	o_4
o_3	o_4	o_3	o_1

Some examples of preferences:

· *Kidney o_2 is preferable than kidney o_4 for patient a_1, so: $o_2 P_{a_1} o_4$.*

· *Patient 1 and donor 4 are compatible, so: $o_4 P_{a_1} o_1$.*

· *Patient 1 and donor 3 are incompatible, so: $o_1 P_{a_1} o_3$.*

Definition 6. Let $a \in A$ be an arbitrary agent. Given a preference P_a, the *weak order of preference* R_a is a transitive, antisymmetric, reflexive, and complete binary relation such that, for every pair of objects $o, o' \in O$, agent a weakly prefers object o' to object o if, and only if, either o and o' are equally preferred objects, or a strictly prefers o to o'; i.e., $oR_a o'$ if and only if one of the following holds:

(i) $o = o'$ (equally preferred), or

(ii) $oP_a o'$ (strictly preferred).

Remark 4. Take $a \in A$.

- P_a is a transitive relation, therefore for all o_i, o_j, and o_k in O, $o_i R_a o_j$ and $o_j R_a o_k$ together imply that $o_i R_a o_k$.
- The relation is antisymmetric, hence if R_a satisfies both $oR_a o'$ and $o' R_a o$ for some $o, o' \in O$, then $o = o'$ (equally preferred).
- R_a is a reflexive relation since $oR_a o$ for all $o \in O$.
- The relation is complete, so every two elements in O are related with each other in some way, i.e. $\forall\ o, o' \in O$ either $oR_a o'$ or $o' R_a o$.

Definition 7. A *profile* P is a list of preference orderings, one for each agent $a \in A$, that is, $P = (P_a)_{a \in A}$.

Definition 8. The 4-tuple (A, O, P, μ) is called *assignment problem.* The set of all assignment problems is denoted by $\mathcal{P}$.

2.2 Mechanisms and their desired properties

The objective is to design a systematic method called mechanism that would suggest for each assignment problem a description (an assignation α) of which object receives each agent.

Let $\mathcal{A}$ be the set of all possible allocations of a given assignment problem.

Definition 9. A *mechanism* is a map $\phi : \mathcal{P} \longrightarrow \mathcal{A}$ such that for each assignment problem $(A, O, P, \mu) \in \mathcal{P}$, $\phi[A, O, P, \mu] \equiv \nu$ is an allocation $\nu : A \longrightarrow O$ in $\mathcal{A}$.

Let us denote $\phi[A, O, P, \mu](a)$ the object assigned to agent a in the assignment problem (A, O, P, μ) by mechanism ϕ.

It would be convenient that the proposal had some good qualities for the benefit of the agents and the proper utilization of the existing objects. To present them, previously some useful definitions will be given.

Definition 10. An allocation $\alpha : A \longrightarrow O$ is said to be *individually rational* (IR) in the assignment problem (A, O, P, μ) if for each agent $a \in A$, $\alpha(a) R_a \mu(a)$.

Definition 11. An allocation $\alpha : A \longrightarrow O$ is said to be *Pareto-efficient* (PE) in the assignment problem (A, O, P, μ) if there does not exist any other assignation ν such that:

(i) $\nu(a) R_a \alpha(a)$ for every $a \in A$, and

(ii) $\nu(a) P_a \alpha(a)$ for some $a \in A$

A mechanism is individually rational or Pareto-efficient if always selects an individually rational or Pareto-efficient allocation, respectively.

Definition 12. A *coalition* is a subset $T \subseteq A$ of agents that cooperate to achieve a common objective.

Definition 13. For a coalition $T \subseteq A$ with $T \neq \emptyset$, a *T-allocation* α^T is defined by a permutation of the set T. An A-allocation is simply called an *allocation*.

Definition 14. A coalition $T \subseteq A$ is *effective* for an allocation α if, for each agent $a \in T$, $\alpha(a) \in \mu(T)$, where $\mu(T) = \{\mu(a) | a \in T\}$.

For any fixed profile, a coalition can improve upon an allocation if the agents of that coalition can trade their initial objects among themselves to make at least one agent obtain a better object without making any other obtain a worse one, such fact is called *domination*.

Definition 15. An allocation ν *strongly dominates* an allocation α if there is some coalition $T \subseteq A$ effective for the allocation ν, such that:

$$\nu(a) P_a \alpha(a) \quad \forall\, a \in T.$$

Thus, ν strongly dominates α if, by trading among themselves, a coalition $T \subseteq A$ could reach a reallocation ν which is strictly preferred by each member of T rather than α.

Definition 16. An allocation ν *weakly dominates* an allocation α if there is some coalition $T \subseteq A$ effective for the allocation ν, such that:

(i) $\nu(a) R_a \alpha(a) \quad \forall\, a \in T$,

(ii) $\nu(a) P_a \alpha(a)$ for some $a \in T$.

Hence, ν weakly dominates α if, by trading among themselves, a coalition $T \subseteq A$ could reach a reallocation ν which is strictly preferred by some member of T rather than α, and at least as preferred as α for each member of T.

Definition 17. The *core* of an assignment problem (A, O, P, μ) is the set of undominated allocations.

The core of an assignment problem can be defined by both strong or weak domination.

Definition 18. An allocation $\alpha : A \longrightarrow O$ is said to be in the *core defined by strong domination* of the assignment problem (A, O, P, μ) if there does not exist any blocking coalition $T \subseteq A$ and any assignation $\nu : A \longrightarrow O$ such that T is effective for ν and ν strongly dominates α, i.e. satisfying:

(i) $\nu(a) \in \mu(T) \quad \forall a \in T$, and

(ii) $\nu(a) P_a \alpha(a) \quad \forall a \in T$.

Definition 19. An allocation $\alpha : A \longrightarrow O$ is said to be in the *core defined by weak domination* of the assignment problem (A, O, P, μ) if there does not exist any blocking subset of agents $T \subseteq A$ and any assignation $\nu : A \longrightarrow O$ such that T is effective for ν and ν weakly dominates α, i.e. satisfying:

(i) $\nu(a) \in \mu(T) \quad \forall a \in T$,

(ii) $\nu(a) R_a \alpha(a) \quad \forall a \in T$, and

(iii) $\nu(a) P_a \alpha(a)$ for some $a \in T$.

By definition, the core defined by weak domination is contained in the core defined by strong domination.

Now, a proposed allocation solution for the assignment problem should satisfy:

(i) Each agent should receive an object at least as good as its initial one; this is, the assignment should be **individually rational**. Otherwise, $\mu(a) P_a \alpha(a)$ and agent a, to whom an incompatible object reception is proposed, could obstruct assignation α. This property is essential if the involvement of the agents in the assignment problem is voluntary.

(ii) The coalition of all traders cannot improve upon the allocation proposed by the mechanism; that is, the assignation needs to be **Pareto-efficient**. This property will be fundamental to ensure the good utilization of the existing objects.

(iii) The assignation proposed by the mechanism can not be blocked by any subset of agents. In other words, it must have the property that there does not exist any subset of agents that could get a better outcome, concerning the assignation proposed, by reallocating between them their initial objects; this means that no coalition of traders can improve upon the proposed allocation. This is, the assignment should be in the **core defined by weak domination** of the assignment problem, and therefore, in the core defined by strong domination.

Lemma 1. Any allocation in the core defined by weak domination is individually rational and Pareto-efficient.

Proof. To prove it, it is just needed to consider in the definition of core by weak domination the possible blocking coalition to consist of each agent itself ($T = \{a\}$, $a \in A$) and the set of all agents ($T = A$), respectively. Suppose α is an allocation in the core defined by weak domination, then:

(i) Assume that α is not IR, then there exists at least an agent $a \in A$ such that $\mu(a) P_a \alpha(a)$. Let us take $T = \{a\}$, μ weakly dominates α via the blocking coalition T. Therefore α must be IR to be in the core defined by weak domination.

(ii) Assume that α is not PE, then there exists some allocation ν such that $\nu(a) R_a \alpha(a)$ for all $a \in A$ and $\nu(a) P_a \alpha(a)$ for some $a \in A$. Therefore, choosing $T = A$ leads us straightforward to a contradiction.

□

Chapter 3

The Top Trading Cycles Algorithm of Gale

3.1 The algorithm

The Top Trading Cycles algorithm of Gale, also known as Gale's TTC algorithm, is a procedure for trading indivisible goods without using money or any other means of exchange. It was developed by David Gale (1921 - 2008), a recognised American mathematician and economist, and published by L. Shapley and H. Scarf (1974) in [39]. The purpose of this algorithm is to find an allocation that assigns to each agent the best available object, taking into account every preference order.

Gale's TTC algorithm solves the assignment problem by steps. In each one proceed as follows: first, a directed graph is constructed in which the nodes are the initial agent-object pairs (a_i, o_i) not assigned in the previous stages; then, a single arc emanates from each node, so that each agent points to its most preferred object among the ones taking part in the current step; finally, the agents corresponding to the nodes belonging to some cycle in the directed graph are assigned the object they point to, respectively. It is important to consider that the algorithm is applied under the hypothesis of the agent set A and object set O being the same size. Agents may report indifference between several objects. In case of indifference, one object should be selected among the equally preferred ones in each case by breaking ties randomly; this could lead into different solutions of the algorithm for the same problem. See the pseudocode in Algorithm 1.

Gale's TTC algorithm identifies cycles successively. Notice that, in each stage, there exists at least one cycle due to the finiteness of the node-set, and if there are more than one cycle they never intersect with each other. Loops are allowed, there can be a node whose agent points to its own object.

The algorithm concludes then, after a finite number of steps.

Algorithm 1: Gale's TTC algorithm.

```
input  : A = {a_1, a_2, ..., a_n}, O = {o_1, o_2, ..., o_n}, P = (P_a)_{a∈A}.
output: An assignment of objects to agents.
N = {1, 2, ..., n};
/* node i represents the initial pair (a_i, o_i)                    */
stages = 0; cycles = 0; loops = 0;
Function inCycle(node)
    if node is not in a cycle nor in a loop, returns FALSE.
     Otherwise, returns TRUE and the vector cycle containing all
     the nodes that form the cycle or loop in which node is.
while N ≠ ∅ do
    stages = stages + 1;
    Construct a directed graph with the nodes that are in N;
    for each element i in N do
        Draw an arc from node i to the node corresponding to a_i's
         first preferential object, among the ones still in P_{a_i}.
    for each node k in N do
        if inCycle(k) is True then
            if length(cycle)==1 then
                loops = loops + 1;
            else
                cycles = cycles + 1;
            for i in cycle do
                o_j is the first preferable object for a_i among the
                 objects remaining unassigned, according to P_{a_i} in P ;
                a_i is assigned to o_j ;
                remove column P_{a_i} from P ;
                remove i from N ;
                remove o_i from each preference ordering in P ;
```

Definition 20. S_k is the set of agents that belong to a cycle and are assigned to objects, and therefore removed from the assignment problem, in the k-th stage of Gale's TTC algorithm. The sets $S_1, S_2, ..., S_K$ are called *top trading cycles*. A top trading cycle may consist of a single trader, and the agents from more than one disjoint cycles can be part of the same top trading cycle.

Definition 21. Let S_k be a top trading cycle. If every agent in S_k belong to the same cycle, S_k can be called *simple top trading cycle*.

Being A the set of agents of the assignment problem (A, O, P, μ), A can be partitioned into a collection of one or more disjoint sets: $A = S_1 \cup S_2 \cup$

$... \cup S_K$, by taking S_1 to be the top trading cycle for A, then taking S_2 to be the top trading cycle for $A \backslash S_1$, S_3 the top trading cycle for $A \backslash (S_1 \cup S_2)$, and so on and so forth will be S_k the top trading cycle for $A \backslash (S_1 \cup S_2 \cup ... \cup S_{k-1})$ following up until there are no agents in $A \backslash (S_1 \cup S_2 \cup ... \cup S_K)$.

The assignment obtained after applying Gale's TTC algorithm to an assignment problem (A, O, P, μ) will be denoted by $\eta : A \longrightarrow O$. By such allocation, each agent obtains the object assigned to it by the trade corresponding to the cycle by which was removed from the problem.

The algorithm would suggest doing the trades described by η. In those in which η coincides with μ (loops) the corresponding agents would stay with their initial object.

By Remark 2, referring to patients in need of a kidney transplantation, the algorithm would propose to perform the transplants described by η, except those in which η coincides with μ, because the corresponding patients would nor receive any kidney and the respective donors would not donate theirs.

Example 3. *Let (A, O, P, μ) be an assignment problem such that $|A| = |O| = 12$, $\mu(a_i) = o_i$ for $i = 1, ..., 12$, and profile P is given by Table 2.*

Table 2: Profile

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}	P_{a_5}	P_{a_6}	P_{a_7}	P_{a_8}	P_{a_9}	$P_{a_{10}}$	$P_{a_{11}}$	$P_{a_{12}}$
o_4	o_{10}	o_{10}	o_2	o_6	o_4	o_2	o_7	o_1	o_6	o_4	o_7
o_3	o_7	o_6	o_1	o_5	o_7	o_1	o_9	o_6	o_3	o_6	o_9
o_9	o_8	o_5	o_6	o_3	o_3	o_5	o_{10}	o_2	o_5	o_1	o_3
o_{13}	o_2	o_9	o_7	o_9	o_2	o_8	o_{11}	o_5	o_7	o_5	o_{10}
o_1	o_{11}	o_1	o_{10}	o_8	o_{11}	o_{10}	o_1	o_4	o_9	o_8	o_6
o_7	o_3	o_7	o_{11}	o_{12}	o_{12}	o_6	o_3	o_{11}	o_{10}	o_2	o_2
o_6	o_4	o_3	o_3	o_{11}	o_5	o_9	o_6	o_9	o_{11}	o_{12}	o_{11}
o_{11}	o_1	o_2	o_{12}	o_4	o_{10}	o_7	o_2	o_3	o_1	o_7	o_5
o_2	o_9	o_{11}	o_5	o_7	o_6	o_4	o_{12}	o_7	o_4	o_9	o_1
o_{10}	o_6	o_8	o_9	o_1	o_8	o_{11}	o_8	o_{10}	o_{12}	o_3	o_{12}
o_8	o_{12}	o_4	o_8	o_{10}	o_9	o_{12}	o_4	o_{12}	o_2	o_{11}	o_4
o_5	o_5	o_{12}	o_4	o_2	o_1	o_3	o_5	o_8	o_8	o_{10}	o_8

Remember that columns represent the preference ranking of each patient over the kidney set in descending order, the items in red cells represent the initial assignment of each patient (μ), and the kidneys in green cells are the preferential graft for each patient.

Let us obtain the final assignment η applying Gale's TTC algorithm. For that, construct a graph with 12 vertices, each one symbolising the corresponding patient-donor pair. Figure 1 represents the graph of the first step, where each recipient points at its most preferred graft.

The only cycle found in the first step is the one represented by permutation (10 6 4 2); in the solution assignment we have: $\eta(a_2) = o_{10}$, $\eta(a_4) = o_2$, $\eta(a_6) = o_4$ and $\eta(a_{10}) = o_6$. Then, nodes 2, 4, 6, and 10 are removed from the assignment problem, and since $A \backslash \{a_2, a_4, a_6, a_{10}\} \neq \emptyset$ step 2 goes on as shown in Table 3 and Figure 2; each patient in $A \backslash \{a_2, a_4, a_6, a_{10}\}$ points now at its preferential kidney in $O \backslash \{o_2, o_4, o_6, o_{10}\}$.

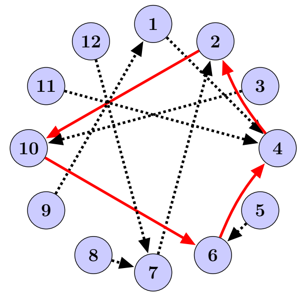

Figure 1: First step

Table 3

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}	P_{a_5}	P_{a_6}	P_{a_7}	P_{a_8}	P_{a_9}	$P_{a_{10}}$	$P_{a_{11}}$	$P_{a_{12}}$
~~o_4~~	~~o_{10}~~	~~o_{10}~~	~~o_2~~	~~o_6~~	~~o_4~~	~~o_2~~	o_7	o_1	~~o_6~~	~~o_4~~	o_7
o_3	~~o_7~~	~~o_6~~	~~o_1~~	o_5	~~o_7~~	o_1	o_9	~~o_6~~	~~o_3~~	~~o_6~~	o_9
o_9	~~o_8~~	o_5	~~o_6~~	o_3	~~o_3~~	o_5	~~o_{10}~~	~~o_2~~	~~o_5~~	o_1	o_3
o_{13}	o_2	o_9	~~o_7~~	o_9	~~o_2~~	o_8	o_{11}	o_5	~~o_7~~	o_5	~~o_{10}~~
o_1	o_{11}	o_1	~~o_{10}~~	o_8	~~o_{11}~~	~~o_{10}~~	o_1	~~o_4~~	~~o_9~~	o_8	~~o_6~~
o_7	o_3	o_7	~~o_{11}~~	o_{12}	~~o_{12}~~	~~o_6~~	o_3	o_{11}	o_{10}	~~o_2~~	~~o_2~~
o_6	o_4	o_3	~~o_3~~	o_{11}	~~o_5~~	o_9	~~o_6~~	o_9	o_{11}	o_{12}	o_{11}
o_{11}	o_1	o_2	~~o_{12}~~	o_4	~~o_{10}~~	o_7	~~o_2~~	o_3	o_1	o_7	o_5
o_2	o_9	o_{11}	~~o_5~~	o_7	o_6	o_4	o_{12}	o_7	o_4	o_9	o_1
o_{10}	o_6	o_8	~~o_9~~	o_1	o_8	o_{11}	o_8	o_{10}	o_{12}	o_3	o_{12}
o_8	o_{12}	o_4	~~o_8~~	o_{10}	o_9	o_{12}	o_4	o_{12}	o_2	o_{11}	o_4
o_5	o_5	o_{12}	o_4	o_2	o_1	o_3	o_5	o_8	o_8	o_{10}	o_8

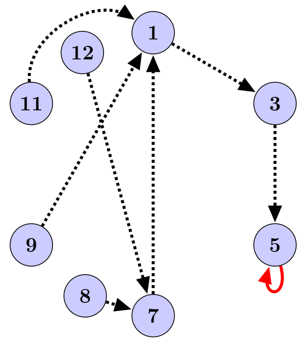

Figure 2: Second step

In this second stage, there is just one loop formed by patient 5 and its initial donor's kidney; a_5 is assigned to its initial graft ($\eta(a_5) = o_5$), and step 3 follows with patients in $A \backslash \{a_2, a_4, a_5, a_6, a_{10}\}$ and their respective initial endowments.

Table 4

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}	P_{a_5}	P_{a_6}	P_{a_7}	P_{a_8}	P_{a_9}	$P_{a_{10}}$	$P_{a_{11}}$	$P_{a_{12}}$
~~o_4~~	~~o_{10}~~	~~o_{10}~~	~~o_2~~	~~o_6~~	~~o_4~~	~~o_2~~	o_7	o_1	~~o_6~~	~~o_4~~	o_7
o_3	~~o_7~~	~~o_6~~	~~o_1~~	o_5	~~o_7~~	o_1	o_9	~~o_6~~	~~o_3~~	~~o_6~~	o_9
o_9	~~o_8~~	~~o_5~~	~~o_6~~	o_3	~~o_3~~	~~o_5~~	~~o_{10}~~	~~o_2~~	~~o_5~~	o_1	o_3
o_{13}	o_2	o_9	~~o_7~~	o_9	~~o_2~~	o_8	o_{11}	~~o_5~~	~~o_7~~	~~o_5~~	~~o_{10}~~
o_1	o_{11}	o_1	~~o_{10}~~	o_8	~~o_{11}~~	~~o_{10}~~	o_1	~~o_4~~	~~o_9~~	o_8	~~o_6~~
o_7	o_3	o_7	~~o_{11}~~	o_{12}	~~o_{12}~~	~~o_6~~	o_3	o_{11}	o_{10}	~~o_2~~	~~o_2~~
o_6	o_4	o_3	~~o_3~~	o_{11}	~~o_5~~	o_9	~~o_6~~	o_9	o_{11}	o_{12}	o_{11}
o_{11}	o_1	o_2	~~o_{12}~~	o_4	~~o_{10}~~	o_7	~~o_2~~	o_3	o_1	o_7	~~o_5~~
o_2	o_9	o_{11}	~~o_5~~	o_7	o_6	o_4	o_{12}	o_7	o_4	o_9	o_1
o_{10}	o_6	o_8	~~o_9~~	o_1	o_8	o_{11}	o_8	o_{10}	o_{12}	o_3	o_{12}
o_8	o_{12}	o_4	~~o_8~~	o_{10}	o_9	o_{12}	o_4	o_{12}	o_2	o_{11}	o_4
o_5	o_5	o_{12}	o_4	o_2	o_1	o_3	o_5	o_8	o_8	o_{10}	o_8

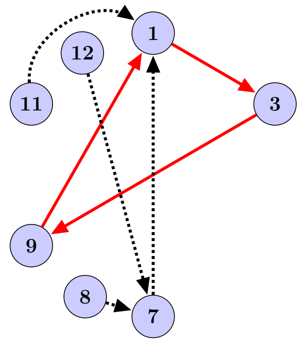

Figure 3: Third step

In step 3, the 3-cycle (1 3 9) is obtained as shown in Figure 3 based on preferences of Table 4. In step 4 the only cycle is (7 8) according to Figure 4 and Table 5.

Table 5

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}	P_{a_5}	P_{a_6}	P_{a_7}	P_{a_8}	P_{a_9}	$P_{a_{10}}$	$P_{a_{11}}$	$P_{a_{12}}$
~~o_4~~	~~o_{10}~~	~~o_{10}~~	~~o_2~~	~~o_6~~	~~o_4~~	~~o_2~~	o_7	~~o_1~~	~~o_6~~	~~o_4~~	o_7
~~o_3~~	~~o_7~~	~~o_6~~	~~o_1~~	o_5	~~o_7~~	~~o_1~~	~~o_9~~	~~o_6~~	~~o_3~~	~~o_6~~	~~o_9~~
~~o_9~~	~~o_8~~	~~o_5~~	~~o_6~~	o_3	~~o_3~~	~~o_5~~	~~o_{10}~~	~~o_2~~	~~o_5~~	~~o_1~~	~~o_3~~
~~o_{13}~~	o_2	~~o_9~~	~~o_7~~	o_9	~~o_2~~	o_8	o_{11}	~~o_5~~	~~o_7~~	~~o_5~~	~~o_{10}~~
o_1	o_{11}	~~o_1~~	~~o_{10}~~	o_8	~~o_{11}~~	~~o_{10}~~	~~o_1~~	~~o_4~~	~~o_9~~	o_8	~~o_6~~
o_7	o_3	~~o_7~~	~~o_{11}~~	o_{12}	~~o_{12}~~	~~o_6~~	~~o_3~~	~~o_{11}~~	o_{10}	~~o_2~~	~~o_2~~
o_6	o_4	o_3	~~o_3~~	o_{11}	~~o_5~~	~~o_9~~	~~o_6~~	o_9	o_{11}	o_{12}	o_{11}
o_{11}	o_1	o_2	~~o_{12}~~	o_4	~~o_{10}~~	o_7	~~o_2~~	o_3	o_1	o_7	~~o_5~~
o_2	o_9	o_{11}	~~o_5~~	o_7	o_6	o_4	o_{12}	o_7	o_4	~~o_9~~	~~o_1~~
o_{10}	o_6	o_8	~~o_9~~	o_1	o_8	o_{11}	o_8	o_{10}	o_{12}	~~o_3~~	o_{12}
o_8	o_{12}	o_4	~~o_8~~	o_{10}	o_9	o_{12}	o_4	o_{12}	o_2	o_{11}	o_4
o_5	o_5	o_{12}	o_4	o_2	o_1	o_3	o_5	o_8	o_8	o_{10}	o_8

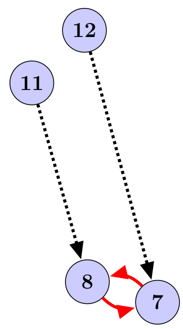

Figure 4: Fourth step

Step 5 proceeds just with nodes 11 and 12. The preference for patient a_{11} is kidney o_{12}, and vice versa; hence, those assignations are carried out and the algorithm ends with the 2-cycle (11 12).

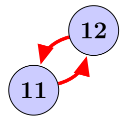

Figure 5: Fifth step

The assignment η in the core obtained by Gale's TTC algorithm can be represented by:

$$\eta = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & a_{11} & a_{12} \\ o_3 & o_{10} & o_9 & o_2 & o_5 & o_4 & o_8 & o_7 & o_6 & o_4 & o_{12} & o_{11} \end{pmatrix}$$

The algorithm would suggest doing the transplants described by η, except those in which η allocates to a patient its initial donor's kidney assigned by μ. This means that every patient will receive a compatible living donor kidney except patient 5.

The top trading cycles will be: $S_1 = \{a_2, a_4, a_6, a_{10}\}$, $S_2 = \{a_5\}$, $S_3 = \{a_1, a_3, a_9\}$, $S_4 = \{a_7, a_8\}$, and $S_5 = \{a_{11}, a_{12}\}$.

3.2 Competitive allocations

Let us denote $p = (p_{o_1}, ..., p_{o_n})$ a *price vector* where $p_{o_i} > 0$ is the *price* assigned to each object o_i. The price can be thought as a non-negative real value useful to determine the accessibility of an object for a patient.

Definition 22. An object o_j is *accessible* for agent a_i in the price vector $p = (p_{o_1}, ..., p_{o_n})$ if $p_{o_j} \leq p_{\mu(a_i)}$.

This is, o_j is accessible for a_i if a_i can obtain o_j after trading its object $\mu(a_i)$.

Example 4. *For the problem in Example 3 the top trading cycles were:* $S_1 = \{a_2, a_4, a_6, a_{10}\}, S_2 = \{a_5\}, S_3 = \{a_1, a_3, a_9\}, S_4 = \{a_7, a_8\},$ *and* $S_5 = \{a_{11}, a_{12}\}$*. Hence, we could take the price vector:*

$$p = (\pi_3, \pi_1, \pi_3, \pi_1, \pi_2, \pi_1, \pi_4, \pi_4, \pi_3, \pi_1, \pi_5, \pi_5),$$

where $\pi_1 > \pi_2 > \pi_3 > \pi_4 > \pi_5 > 0$*. The accessible kidneys for patient* a_8 *will be the ones initially corresponding to patients in* S_4 *and* S_5 *(i.e.,* $o_7, o_8, o_{11},$ *and* o_{12}*), and the preferable one for it among them will be* o_7*, the one allocated by Gale's TTC algorithm.*

Definition 23. A pair (p, α) where p is a price vector and α is an allocation is an *efficiency equilibrium* if, for every agent $a_i \in A$, $\alpha(a_j) P_{a_i} \alpha(a_i)$ implies $p_{o_j} > p_{o_i}$.

If (p, α) is an efficiency equilibrium, then the allocation α assigns to each agent a the best object it could purchase (or access to) at the prices p.

Definition 24. An allocation α is *efficient* if there exists a price vector p such that (p, α) is an efficiency equilibrium.

Definition 25. An efficiency equilibrium (p, α) is named *competitive equilibrium* if for all $a_i \in A$:

(i) $p_{\alpha(a_i)} \leq p_{\mu(a_i)}$, and

(ii) $\alpha(a_i) R_{a_i} o_j \; \forall \, o_j$ such that $p_{o_j} \leq p_{o_i}$.

Definition 26. An allocation α is a *competitive allocation* if there exists a price vector p such that (p, α) is a competitive equilibrium.

Alvin E. Roth and Andrew Postlewaite (1977) stated in [36] the following lemma:

Lemma 2. Any competitive allocation can be thought of as resulting from the method of Top Trading Cycles.

Proof. Let $S_1 \subseteq A$ be the set of agents whose initial objects are priced highest among all the objects taking part in the problem. Then, all agents in S_1 must be getting their most preferred object in the whole set of objects O. Furthermore, they must only be trading among themselves, since if any agent is assigned the object of some other agent outside S_1, then there must be outside of S_1 an agent a obtaining the initial endowment of some agent in S_1, which has a higher price than its initial object $\mu(a)$. But, this is clearly impossible in a competitive allocation. For the same reason, two agents in S_1 can not prefer most the same object.

Now, if $S_2 \subseteq A \backslash S_1$ is the set of agents whose initial objects are priced highest among the objects owned by the agents in $A \backslash S_1$. Then the agents in S_2 must be receiving their most preferred objects among those belonging to agents in $A \backslash S_1$; that, arguing as before, must belong particularly to agents in S_2.

Proceeding this manner S_1 will be a top trading cycle for A, S_2 a top trading cycle for $A \backslash S_1$, ..., S_k a top trading cycle for $A \backslash (S_1, S_2, ..., S_{k-1})$, ... Until $A \backslash (S_1 \cup S_2 \cup ... \cup S_K) = \emptyset$ for some $K > 0$; hence, $A = S_1 \cup S_2 \cup ... \cup S_K$. □

The idea of the following theorem was stated by Lloyd Shapley and Herbert Scarf (1974) in [39] and stated as a theorem by Jun Wako (1984) in [45]:

Theorem 3. The set of allocations generated by the method of Top Trading Cycles coincides with the set of competitive allocations.

Proof. The proof is deduced from Lemma 2 and the following argument. Any allocation α given by Gale's TTC algorithm is competitive since agents find their most preferred object, among their available options set, in the cycle in which they are removed from the problem. In other words, there exists a set of competitive prices for an allocation α given by Gale's TTC algorithm. Just assign arbitrary prices $\pi_1 > \pi_2 > ... > \pi_k > 0$ to the initial objects of agents in $S_1, S_2, ..., S_k$, respectively. Then, an agent $a \in S_j$ can trade its initial object at price π_j; hence, it cannot access the objects that initially belong to agents in $S_1, S_2, ..., S_{j-1}$. Therefore, the agent's benefit is maximized if obtains the initial object of its cyclic successor in S_j, which has exactly the same price as its initial object.

□

For $T \subseteq A$ and $a_i \in T$, let us denote $B(T, a_i)$ as the set of most preferred objects in $\mu(T)$ for agent a_i, i.e. $B(T, a_i) = \{o \in \mu(T) \mid oR_{a_i}\mu(a_j), a_j \in T\}$.

Definition 27. Let T be a non-empty subset of A, and ν any allocation. A simple *top trading cycle for T induced by ν* is a non-empty subset S of T, whose s members can be indexed in a cyclic order, $S = \{a_{\sigma(1)}, a_{\sigma(2)}, ..., a_{\sigma(s)} = a_{\sigma(0)}\}$, in the following way: for each $a_{\sigma(k)}$, if $\nu(a_{\sigma(k)}) \in B(T, a_{\sigma(k)})$, then $\nu(a_{\sigma(k)}) = \mu(a_{\sigma(k+1)})$, and if $\nu(a_{\sigma(k)}) \notin B(T, a_{\sigma(k)})$ then $a_{\sigma(k+1)} = a_j$, where a_j is some agent such that $\mu(a_j) \in B(T, a_{\sigma(k)})$.

Let us introduce the next two results published by Jun Wako (1984) in [45]:

Lemma 4. Let T be a non-empty subset of A and ν any allocation. A top trading cycle for T induced by ν is a top trading cycle for T.

Proof. Notice that for any non-empty set $T \subseteq A$, and any allocation ν, there exists at least one top trading cycle for T induced by ν. Let us consider an allocation ν. The top trading cycle(s) for $T \subseteq A$, $T \neq \emptyset$, induced by allocation ν can be reached as follows:

Step 1: Choose any agent $a_{\sigma(1)} \in T$.

Step 2: If $\nu(a_{\sigma(1)}) \in B(T, a_{\sigma(1)})$ then let $\nu(a_{\sigma(1)}) = \mu(a_{\sigma(2)})$.
If $\nu(a_{\sigma(1)}) \notin B(T, a_{\sigma(1)})$ then choose any $\mu(a_k) \in B(T, a_{\sigma(1)})$ and let $a_{\sigma(2)} = a_k$.

Step 3: If $\nu(a_{\sigma(2)}) \in B(T, a_{\sigma(2)})$, then let $\nu(a_{\sigma(2)}) = \mu(a_{\sigma(3)})$.
If $\nu(a_{\sigma(2)}) \notin B(T, a_{\sigma(2)})$, then choose any $\mu(a_t) \in B(T, a_{\sigma(2)})$ and let $a_{\sigma(3)} = a_t$.

Proceed in the same manner until an agent that was chosen in a previous stage is chosen. Then, for T being a finite set, the method must stop in a finite number of steps less or equal than $|T| + 1$. Hence there exists some integer m, $1 \leq m \leq |T|$, and some integer q, $1 \leq q \leq m$, such that the agents of the subset $\{a_{\sigma(q+1)}, ..., a_{\sigma(m)}, a_{\sigma(m+1)} = a_{\sigma(q)}\}$ form a cycle and are a top trading cycle for T (or at least a subset of a top trading cycle for T). □

Theorem 5. The core defined by weak domination is included on the set of competitive allocations.

Proof. Let us proof that every allocation in the core is competitive. Bear in mind that the core is defined by weak domination; then, the indifference among objects is allowed.

If the core is empty, the theorem is trivial. Assume that the core is non-empty. Let ν^* be any allocation in the core. It is sufficient to show that ν^* is a competitive allocation.

Recall that $B(T, a_i) = \{o \in \mu(T) \mid oR_{a_i}\mu(a_j),\ a_j \in T\}$, for $T \subseteq A$ and $a_i \in T$. Let $S_1, S_2, S_3, ..., S_K$ be the top trading cycles induced by ν^*. Let ν be the allocation resulting from carrying out the trades within each cycle in S_k for $k = 1, 2, ..., K$. Then ν is a competitive allocation by Theorem 3. It will be shown that $\nu(a_i) = \nu^*(a_i)$ for all $a_i \in S_k$, by induction on k.

First we will show that $\nu(a_i) = \nu^*(a_i)\ \forall\ a_i \in S_1$. By reductio ad absurdum suppose that there exists an agent $a_t \in S_1$ such that $\nu(a_t) \neq \nu^*(a_t)$. Since $\nu(a_t)$ is the object assigned by the top trading cycle S_1 for A induced by ν^*, the previous inequality implies that $\nu^*(a_t) \notin B(A, a_t)$. Now,

$$\left.\begin{array}{l} \nu^*(a_t) \in \mu(A), \\ \nu(a_t) \in B(A, a_t) \end{array}\right\} \xrightarrow{\nu^*(a_t) \notin B(A, a_t)} \nu(a_t) P_{a_t} \nu^*(a_t), \qquad (1)$$

and $\forall a_i \in S_1 \backslash \{a_t\}$, as it can be either $\nu(a_i) = \nu^*(a_i)$ or $\nu(a_i) \neq \nu^*(a_i)$:

$$\left.\begin{array}{l}\nu^*(a_i) \in \mu(A),\\ \nu(a_i) \in B(A, a_i)\end{array}\right\} \Longrightarrow \nu(a_i) R_{a_i} \nu^*(a_i). \tag{2}$$

Additionally, by how ν is defined, $\mu(S_1) = \{\nu(a_i) | a_i \in S_1\}$, i.e. S_1 is effective for ν. (3)

Thus, from (1), (2), and (3), ν^* is weakly dominated by ν via S_1, then ν^* does not belong to the core, leading us to a contradiction. Hence, $\forall a_i \in S_1$, $\nu(a_i) = \nu^*(a_i)$.

Suppose now by the induction hypothesis that for all S_q, $q = 1, 2, ..., k-1$: $\nu(a_i) = \nu^*(a_i) \; \forall \; a_i \in S_q$.

And let us prove that $\nu(a_i) = \nu^*(a_i)$ for all $a_i \in S_k$. By reduction ad absurdum suppose that there exists an agent $a_{t'} \in S_k$ such that $\nu(a_{t'}) \neq \nu^*(a_{t'})$. Once again, since $\nu(a_{t'})$ is the object assigned to $a_{t'}$ by the top trading cycle S_k for $A \backslash (S_1 \cup S_2 \cup ... \cup S_{k-1})$ induced by ν^*, $\nu(a_{t'}) \neq \nu^*(a_{t'})$ implies $\nu^*(a_{t'}) \notin B(A \backslash \cup_{q=1}^{k-1} S_q, a_{t'})$.

By the hypothesis of the induction, each S_q, $q = 1, 2, ..., k-1$, is effective for ν; hence, $\cup_{q=1}^{k-1} S_q$ is effective for ν. Moreover, since for each agent $a_i \in S_q$, where $q = 1, 2, ..., k-1$, $\nu(a_i) = \nu^*(a_i)$ holds, $\cup_{q=1}^{k-1} S_q$ is also effective for ν^*. And, for ν^* being an allocation, the complement $A \backslash \cup_{q=1}^{k-1} S_q$ also must be effective for ν^*, i.e. $\{\nu^*(a_j) | a_j \in A \backslash \cup_{q=1}^{k-1} S_q\} = \mu(A \backslash \cup_{q=1}^{k-1} S_q)$.

Consequently, $\nu^*(a_{t'}) \in \mu(A \backslash \cup_{q=1}^{k-1} S_q)$ since $a_{t'} \in S_k$ and S_k is a subset of $A \backslash \cup_{q=1}^{k-1} S_q$. Thus:

$$\left.\begin{array}{l}\nu^*(a_{t'}) \in \mu(A \backslash \cup_{q=1}^{q=k-1} S_q),\\ \nu(a_{t'}) \in B(A \backslash \cup_{q=1}^{q=k-1} S_q, a_{t'})\end{array}\right\} \Longrightarrow \nu(a_{t'}) P_{a_{t'}} \nu^*(a_{t'}), \tag{4}$$

and $\forall a_j \in S_k \backslash \{a_{t'}\}$:

$$\left.\begin{array}{l}\nu^*(a_j) \in \mu(A - \cup_{q=1}^{q=k-1} S_q),\\ \nu(a_j) \in B(A - \cup_{q=1}^{q=k-1} S_q, a_j)\end{array}\right\} \Longrightarrow \nu(a_j) R_{a_j} \nu^*(a_j). \tag{5}$$

Additionally, by how ν is defined, $\mu(S_k) = \{\nu(a_i) | a_i \in S_k\}$. (6)

Hence, from (4), (5), and (6), ν^* is weakly dominated by ν via S_k, therefore ν^* is not in the core, leading us to a contradiction. Consequently $\nu(a_i) = \nu^*(a_i)$ for all $a_i \in S_k$.

In conclusion $\nu^* = \nu$, and therefore, ν^* is a competitive allocation. □

Alvin E. Roth and Andrew Postlewaite (1977) proved also the next lemma in [36]:

Lemma 6. If no agent is indifferent between any objects, then a competitive allocation weakly dominates any other allocation.

Proof. If η is any competitive allocation, by Lemma 2, it can be thought of as resulting from the method of Top Trading Cycles; this is, by trading among top trading cycles $S_1, S_2, ..., S_K$. Let ν be any other allocation. If $\nu(a) \neq \eta(a)$ for each $a \in S_1$, η strongly dominates ν (and therefore, weakly too) via coalition S_1 since S_1 is effective for η, and η gives each agent on S_1 its most preferred object (bear in mind that there is no indifference allowed). In the same way, η weakly dominates ν if $\nu(a) \neq \eta(a)$ just for some $a \in S_1$.

On the contrary, if $\nu(a) = \eta(a)$ for each agent $a \in S_1$, but $\nu(a) \neq \eta(a)$ for each agent $a \in S_2$, η weakly dominates ν via $S_1 \cup S_2$, since $S_1 \cup S_2$ is effective for η, $\eta(a) R_a \nu(a)$ for each $a \in S_1 \cup S_2$, and $\eta(a) P_a \nu(a)$ for some $a \in S_1 \cup S_2$. Exactly $\eta(a) P_a \nu(a)$ for each $a \in S_2$ since there is no indifference, both η and ν give agents in S_1 their most preferred object, and η gives the agents in S_2 their most preferred object from $O \backslash \eta(S_1)$. In the same way, η weakly dominates ν if $\nu(a) \neq \eta(a)$ just for some $a \in S_2$.

Following with the previous strategy, it can be shown that if no agent has indifferences between any objects, η weakly dominates any other allocation ν. □

3.3 Non-emptiness of the core

The core defined by strong domination is always non-empty, but the core defined by weak domination may be empty. Nevertheless, if no agent is indifferent between any of the objects on the problem , then the core defined by weak domination is always non-empty; in this case, the core coincides with the unique competitive allocation (this last fact will be studied at the end of section 3.5).

Alvin E. Roth and Andrew Postlewaite (1997) stated in [36] that the core defined by strong domination is always non-empty, i.e. whether indifference is allowed or not.

Lemma 7. The core defined by strong domination always contains the set of competitive allocations, which is itself non-empty, and can contain several allocations; so, the core as defined is always non-empty.

Proof. Let α be a competitive allocation. Then by Lemma 2 we know that it can be thought as resulting from the Top Trading Cycles algorithm. Let then $S_1, S_2, ..., S_K$ be the top trading cycles associated to α, and notice that they form a partition of A.

On the other hand, by the definition of competitive allocation we know that there exists a price vector $p = (p_{o_1}, p_{o_2}, ..., p_{o_m})$ such that for all $a_i \in A$:

(i) $p_{\alpha(a_i)} \leq p_{\mu(a_i)}$, and

(ii) $\alpha(a_i) R_{a_i} o_j \; \forall \; o_j$ s.t. $p_{o_j} \leq p_{o_i}$.

By reductio ad absurdum, suppose there exists another allocation ν ($\nu \neq \alpha$) that strongly dominates α via some coalition of agents $T \subseteq A$.

Take now an agent $a_j \in S_k \cap T$ for some $k \in \{1, 2, ..., K\}$. For S_k being a top trading cycle, lets say that, $\forall \; a \in S_k$, $p_{\alpha(a)} = p_{\mu(a)} = \pi_k$ being π_k such that $\pi_1 > \pi_2 > ... > \pi_k > ... > \pi_K$. Furthermore, $a_j \in T$, hence $\nu(a_j) P_{a_j} \alpha(a_j)$ by the definition of strong domination. Therefore, by the properties of the competitive allocation, $p_{\nu(a_j)} > p_{\alpha(a_j)}$. Thus, $p_{\nu(a_j)} = \pi_r$ for some $r < k$, and a_j is an agent in S_r. But $a_j \in S_k$ and $a_j \in S_r$ for $r < k$ lead us to a contradiction. $\square$

In Example 5 will be seen how the core defined by weak domination can be empty if indifference between objects is allowed. Lemma 4 and Theorems 3 and 5 will be utilized. This example is inspired in the one studied by Jun Wako (1991) in [46].

Example 5. *Given an assignment problem with 3 pairs agent-object and the following preferences of the agents:*

(i) $o_2 P_{a_1} o_3 I_{a_1} o_1$

(ii) $o_1 I_{a_2} o_3 P_{a_2} o_2$

(iii) $o_2 P_{a_3} o_1 P_{a_3} o_3$

Table 6: Preferences.

R_{a_1}	R_{a_2}	R_{a_3}
o_2	$o_1 = o_3$	o_2
$o_1 = o_3$		o_1
	o_2	o_3

where $o_i I_{a_k} o_j$ means that agent a_k is indifferent between objects o_i and o_j.

By Gale's TTC algorithm, two possible competitive allocations are obtained:

$$\alpha_1 = \begin{pmatrix} a_1 & a_2 & a_3 \\ o_2 & o_1 & o_3 \end{pmatrix} \quad \text{and} \quad \alpha_2 = \begin{pmatrix} a_1 & a_2 & a_3 \\ o_1 & o_3 & o_2 \end{pmatrix},$$

but, for each assignment, a coalition of agents effective for some allocation α can be found such that α weakly dominates it:

(1) α_1 is weakly dominated via coalition $T = \{a_2, a_3\}$ by allocation ν in which $\nu(a_2) = o_3$ and $\nu(a_3) = o_2$, since T is effective for ν and:

(i) $\nu(a_2) R_{a_2} \alpha(a_2)$, $\nu(a_3) R_{a_3} \alpha(a_3)$, *and*

(ii) $\nu(a_3) P_{a_3} \alpha(a_3)$

(2) In a similar way can be seen that α_2 is weakly dominated via coalition $T' = \{a_1, a_2\}$ by allocation ν' in which $\nu'(a_1) = o_2$ and $\nu'(a_2) = o_1$.

Therefore, neither α_1 nor α_2 belong to the core defined by weak domination. By Lemma 2 it is known that there is not other possible competitive allocation. Hence, the core defined by weak domination is empty by Theorem 5.

The TTC algorithm from Gale was indeed first presented in [39] by L. Shapley and H. Scarf (1974) as an alternative proof to their original one, given in the same document, for the following theorem:

Theorem 8. In every assignment problem with no indifference between objects, the core defined by weak domination is always non-empty.

Proof. Let η be the allocation obtained by Gale's TTC algorithm for the assignment problem (A, O, P, μ). Remember that for η to be in the core by weak domination, there should not exist any coalition of agents which, by trading among its members, could allocate to each one an object which they prefer at least as much as the one which they receive by assignment η, at least one of them receiving an object that strictly prefers rather than the one allocated by η.

Let us prove that η is in the core defined by weak domination of the problem. To this effect, suppose that to solve problem (A, O, P, μ), K steps of Gale's TTC algorithm are needed, then there will be K disjoint sets $S_1, S_2, ..., S_K$, one for each stage $1, 2, ..., K$, respectively. Recall that in S_j there could be more than one cycle or loop ($j = 1, 2, ..., K$). Notice that K stages can be considered:

(1) No agent in S_1 can belong to a blocking coalition for allocation η, strictly preferring such reallocation, since each of them has already been assigned to its best object in O.

(2) No agent in S_2 can belong to a blocking coalition for allocation η, strictly preferring such reallocation, since each of them has already been assigned to its best object in $O \backslash \eta(S_1)$.

Let us show it by reductio ad absurdum. Take $a \in S_2$ with $\eta(a) = o$. Suppose that via some coalition $T \subseteq A$ for which $a \in T$, by some allocation ν, the agent a could be assigned an object o' that strictly prefers, i.e. such that $\nu(a)P_a\eta(a)$; that is $o'P_ao$. Assume also that $\nu(b)R_b\eta(b)$ and $\nu(b) \in \mu(T)\ \forall\ b \in T$.

Since o is the best object for a in $O\backslash\eta(S_1)$, o' can not be in that subset. Therefore, $o' \in \eta(S_1)$, and thus, it is the initial endowment of some agent $a' \in S_1 \cap T$. Hence, $\eta(a') \in \eta(S_1)$ too. Considering that $\nu(b)R_b\eta(b)\ \forall\ b \in T$ and $\eta(a')$ is the overall best object for a' then $\nu(a') = \eta(a')$, that is at the same time the initial endowment of another agent $a'' \in S_1 \cap T$.

Following this way we arrive to a point in which $\nu(a^{(j)}) = \eta(a^{(j)}) = o'$ for some $a^{(j)} \in S_1 \cap T$. But $o' = \nu(a)$, and since each object can be assigned to an only agent, and:

$$\left.\begin{array}{l}\nu(a^{(j)}) = o' \\ \nu(a) = o'\end{array}\right\} \Longleftrightarrow a = a^{(j)}.$$

But $a \in S_2$ and $a^{(j)} \in S_1$, and since those subsets are disjoint, there is a contradiction.

(K) No agent in S_K can belong to a blocking coalition for allocation η, strictly preferring such reallocation, since each of them has already been assigned to its best object in $O\backslash(\eta(S_1) \cup \eta(S_2) \cup ... \cup \eta(S_{K-1}))$. An argument analogous to the one utilised before could be used to prove it.

In conclusion, there is no agent a in some coalition T such that by some allocation ν could satisfy $\nu(a)P_a\eta(a)$. And therefore, η is an allocation in the core defined by weak domination of the assignment problem (A, O, P, μ) since it can not be blocked by any subset of agents. □

Hence, Gale's TTC algorithm always selects an allocation in the core defined by weak domination, and in consequence, no assignment problem has an empty core defined by weak domination.

Alvin E. Roth and Andrew Postlewaite (1997) proved in [36] that, under no indifference among objects, there is no allocation in the core other than the one given by Gale's TTC algorithm:

Theorem 9. If no agent is indifferent between any objects, the core defined by weak domination of each assignment problem contains exactly one allocation.

Proof. Let η be the allocation obtained by Gale's TTC algorithm for the assignment problem (A, O, P, μ) and consider ν another allocation for the same problem $(\nu \neq \eta)$. It wants to be proved that ν can not be in the core defined by weak domination of the assignment problem.

Let k be the first stage in which there is an agent $b \in S_k$ such that $\nu(b) \neq \eta(b)$ (if there are more than one agents satisfying this property, b is chosen arbitrarily among them). Then $\nu(a) = \eta(a)\ \forall a \in S_1 \cup S_2 \cup ... \cup S_{k-1}$.

Since η allocates to each agent $a \in S_k$ its preferred object in the available object set $O \backslash (\eta(S_1) \cup \eta(S_2) \cup ... \cup \eta(S_{k-1}))$, and $\nu(a)$ must belong to that same set, $\nu(a)$ can be at most as preferable, for agent a, as $\eta(a)$. Hence, it will be $\eta(a) R_a \nu(a)$. And besides, from Gale's TTC algorithm, $\forall a \in S_k$, $\eta(a) \in \mu(S_k)$. Recall that there is no indifferece between objects for each agent; thus, by definition of η, since $\nu(b) \neq \eta(b)$, $\eta(b) P_b \nu(b)$.

Therefore, the subset S_k of agents blocks allocation ν; since they can improve concerning the allocation proposed, by trading among themselves the objects initially assigned by μ (via allocation η). In conclusion, ν can not be in the core defined by weak domination of the assignment problem (A, O, P, μ). □

Henceforth, strict preferences in which there is no indifference between objects will be utilized in the coming sections. From this point on, the core defined by weak domination will be denoted just 'core', and weak domination just 'domination'.

3.4 Manipulability of Gale's TTC mechanism

The preference ordering P_a of agent a is who bears the responsibility for the choice of the object that a receives by allocation η resulting from Gale's TTC algorithm over problem (A, O, P, μ). For this reason, an agent might wonder if, displaying a different preference, it could end up receiving a better object than the one obtained by Gale's TTC algorithm. In the following it will be demonstrated that the TTC algorithm encourages the agents to reveal their true preferences.

Given a profile P and a new preference P'_a for agent $a \in A$, let us denote (P_{-a}, P'_a) the profile obtained when replacing in P the preference P_a with P'_a.

Definition 28. A mechanism $\phi : \mathcal{P} \longrightarrow \mathcal{A}$ is *manipulable* if there exists an assignment problem (A, O, P, μ), an agent $a \in A$ and a preference P'_a such that: $\phi[A, O, (P_{-a}, P'_a), \mu](a)\ \ P_a\ \ \phi[A, O, P, \mu](a)$.

In a manipulable mechanism, agent a receives a better object (according to P_a) by declaring P'_a instead of P_a. In that case, it is said that agent a manipulates ϕ in the assignment problem (A, O, P, μ) by declaring P'_a.

Definition 29. A mechanism that is not manipulable is said to be *strategy-proof* (SP).

A strategy-proof mechanism's agent can not obtain an object more preferred than the one obtained when revealing his true preference, by misstating his preference while others stay with his trustful one. Hence, the agents always have incentives to declare their true preference in such a mechanism. The strategy-proofness of the mechanism ensures the use of proper information, and hence the mechanism's suggestion satisfies the rest of ideal properties such as individual rationality and Pareto-efficiency.

Alvin E. Roth (1982) established in [34] that the mechanism of the core is strategy-proof, i.e. Gale's TTC algorithm, as a $\varphi : \mathcal{P} \longrightarrow \mathcal{A}$ mechanism, is not manipulable. To prove the result, first let us consider lemmas 10, 11, and 12, where the notation summary is as follows:

P: fixed preference profile (without indifferences among objects).

TTC_P: Gale's TTC algorithm over the problem (A, O, P, μ).

G_k^P: Graph at the k-th stage of TTC_P.

P': Preference profile which differs from P only in the report of a single (fixed) agent a_i who reports P'_{a_i} instead of P_{a_i}.

S_k^P: set of agents that being part of a cycle (or loop) in stage k of TTC_P are allocated to an object and removed from the problem in the k-th step of the algorithm.

Lemma 10. Let $C = (n_1, n_2, ..., n_m)$ be a chain in the graph G_k^P, and $r > k$, then C is a chain in G_r^P if, and only if, n_m is a node of G_r^P.

Proof. See Appendix A. □

Lemma 11. Take P and P' as defined above, and let k and k' be the stages at which agent a_i is removed from the market in TTC_P and $TTC_{P'}$, respectively. Then the graphs G_l^P and $G_l^{P'}$ have the same cycles for $1 \leq l \leq min\{k, k'\} - 1$, and the same nodes for $1 \leq l \leq min\{k, k'\}$.

Proof. See Appendix A. □

Lemma 12. Let P'' be a preference profile which differs from P' only in the report of agent a_i, where P''_{a_i} is any preference such that $\eta'(a_i) P''_{a_i} \eta'(a_q)$ for all $a_q \neq a_i$ (being η' the allocation resulting from $TTC_{P'}$). Then if η'' is the resulting allocation from $TTC_{P''}$, $\eta''(a_i) = \eta'(a_i)$.

Proof. See Appendix A. □

Theorem 13. The mechanism of the core is strategy-proof.

Proof. Let us consider:

> $\varphi : \mathcal{P} \longrightarrow \mathcal{A}$: mechanism that selects for each assignment problem the only allocation in the core (the correspondent one obtained by Gale's TTC algorithm).
>
> $(A, O, P, \mu) \in \mathcal{P}$: a fixed assignment problem.
>
> $\varphi[A, O, P, \mu] = \eta$: allocation obtained from Gale's TTC algorithm's mechanism on problem (A, O, P, μ).

Consider P and P' as defined above, and let η and η' be the allocations resulting from TTC_P and $TTC_{P'}$, respectively, with $\eta'(a_i) = \mu(a_j)$; where μ is the initial endowment, and a_i is the agent that reports P'_{a_i} instead of its real preference ordering. Then, to prove the theorem, it is enough to show that no P'_{a_i} exists for which $\eta'(a_i)P_{a_i}\eta(a_i)$, i.e. if P_{a_i} is agent a_i's sincere preference, reporting P'_{a_i} instead of P_{a_i} the agent a_i will not get a preferred outcome.

Let k and k' be the steps of TTC_P and $TTC_{P'}$, respectively, at which agent a_i is removed from the market (i.e., $a_i \in S_k^P$ and $a_i \in S_{k'}^{P'}$). The node corresponding to the pair agent-object (a_i, o_i) will be represented as i for each $i = 1, 2, ..., n$, being n the number of couples participating in the problem.

Assume that either:

(i) $\eta'(a_i)P_{a_i}\eta(a_i)$, or

(ii) $\eta'(a_i) = \eta(a_i)$.

Let us conclude that (ii) holds. By reductio ad absurdum suppose that (i) holds. Notice that (i) reflects that agent a_i would be receiving a strictly more preferable object (according to the truthful preference P_{a_i}) by revealing P'_{a_i} as preference rather than its real preference P_{a_i}. On the other hand, (ii) indicates that, at the end, a_i would be receiving the same object revealing either preference P'_{a_i} or P_{a_i}.

By Lemma 12, to consider P'_{a_i} that ranks $\eta(a_i)$ first (i.e., $\eta(a_i)P'_{a_i}\eta(a_j)$ $\forall$ $a_j \neq a_i$), implies straightforwardly $\eta'(a_i) = \eta(a_i)$. That is, P'_{a_i} ranks $\eta'(a_i)$ first, i.e. $\eta'(a_i)P'_{a_i}\eta'(a_j)$ $\forall$ $a_j \neq a_i$.

For any other construction of P'_{a_i}, continue as follows. If $k' \geq k$, then Lemma 11 states that, for $1 \leq l \leq k$, G_l^P and $G_l^{P'}$ have the same nodes. Suppose now that $\eta'(a_i)P_{a_i}\eta(a_i)$ (hypothesis (i), $\eta'(a_i) \neq \eta(a_i)$), then by Lemma 10, and considering that the arc (i, j') must be an arc of G_s^P for $s < k$ being $\eta'(a_i) = o_{j'}$, then (i, j') must be an arc of G_k^P too because j' is a node of G_k^P. As $\eta(a_i) = o_j$, the arc (i, j) is obviously an arc in G_k^P too (is the arc that assigns o_j to agent a_i in stage k and removes it from the problem in that same step). Hence, since $j \neq j'$ and there can not be two arcs emanating from i, a contradiction is noticed. Therefore, it must be $\eta'(a_i) = \eta(a_i)$ and $k = k'$.

Now, if $k' \leq k$, once again by Lemma 11, for $1 \leq l \leq k'$ G_l^P and $G_l^{P'}$ graphs have the same nodes. Let $C = (j' = n_1, n_2, ..., n_m = i)$ be the cycle that forms at step k' of $TTC_{P'}$ (it is the cycle that will remove a_i from the problem at stage k' of $TTC_{P'}$). Unless maybe the arc emanating from i, $G_{k'}^P$ and $G_{k'}^{P'}$ will have the same arcs (consider that a_i was the only agent who reported a different preference for P'); therefore, unless maybe (i, j') all the other arcs of C will be in $G_{k'}^P$ (and in $G_{k'}^{P'}$, obviously). So, C forms a chain in $G_{k'}^P$, and by Lemma 10, since $k \geq k'$ and i is a node in G_k^P, then C forms a chain also in G_k^P. Take once again $\eta'(a_i)P_{a_i}\eta(a_i)$, i.e. $o_{j'}P_{a_i}o_j$, from C being a chain in G_k^P, it is known that j' is a node of G_k^P. Hence, the arc emanating from i would point j' due to the strict preference, and (i, j) could not be an arc in G_k^P, which gives a contradiction for $\eta(a_i) = o_j$. Therefore, it must be $\eta'(a_i) = \eta(a_i)$ and $k = k'$. □

3.5 Desirable properties of Gale's TTC algorithm

The idea of core may be too demanding for the implementation of Gale's TTC algorithm to the assignment problem of living donor kidney exchange due to the fact that it could be difficult for an agent to realize that by becoming part of a coalition of agents, and redistributing their initial objects among themselves, it would end up receiving a strictly preferred object than the one obtained by Gale's TTC algorithm, and the rest of participants in the coalition receiving an object at least as good for them as the one obtained in the first case. Additionally, it would not be feasible to perform the paired transplants outside the institution that rules the program.

Nevertheless, in the general scope of this dissertation, it is sensible to require that the mechanism is individually rational and efficient, and that encourages the participants to state their truthful preferences. Jinpeng Ma (1994) proved in [16] that a mechanism satisfies individual rationallity, Pareto-efficiency and strategy-proofness if, and only if, it is the core mechanism. In order to proof that result, lemmas 14 to 18 will be presented and proved before.

Definition 30. Given the profile P and two allocations $\alpha, \omega : A \longrightarrow O$ for the assignment problem (A, O, P, μ), let us define $J(\alpha, \omega, P)$ the agents that strictly prefer allocation α to ω according to preferences in profile P, i.e. $J(\alpha, \omega, P) := \{a \in A : \alpha(a) P_a \omega(a)\}$. Notice that $J(\alpha, \omega, P)$, $J(\omega, \alpha, P)$, and $A \backslash (J(\alpha, \omega, P) \cup J(\omega, \alpha, P))$ form a partition of A.

For the next lemmas take a profile P honestly declared by the agents, and assume that φ is the mechanism of the core and ϕ is an individually rational, Pareto-efficient and strategy-proof mechanism; as they will be used later to prove Theorem 19.

Lemma 14. Let α, ω be Pareto-efficient with respect to profile P, and suppose $\alpha \neq \omega$. Then $J(\alpha, \omega, P)$ is not empty.

Proof. See Appendix A. □

Lemma 15. Let η be the allocation in the core of assignment problem (A, O, P, μ), and an allocation ν individually rational and Pareto-efficient w.r.t. profile P, being $\eta \neq \nu$. Then, exists $a \in J(\eta, \nu, P)$ such that

$$\eta(a) P_a \nu(a) P_a \mu(a).$$

Proof. See Appendix A. □

Let us consider the following notation: for a fixed problem (A, O, P, μ), $\varphi_P(a_i) := \varphi[A, O, P, \mu](a_i)$.

For a profile P, let us define T_P as the agents that strictly prefer (according to profile P) some object o to the initial one, being o strictly less preferred than the one assigned by φ, i.e. $T_P = \{a \in A : \exists\, o \in O \text{ s.t. } \varphi_P(a) P_a o P_a \mu(a)\}$ and the profile of preferences $P' = (P'_{a_1}, P'_{a_2}, ..., P'_{a_n})$ as follows:

$$P'_{a_i} = \begin{cases} (..., \varphi_P(a_i), \mu(a_i), ...) & \text{if } a_i \in T_P \\ P_{a_i} & \text{if } a_i \in A \backslash T_P \end{cases} \tag{3.1}$$

Recall:

$$\begin{array}{ll} P_{-a_i} := (P_{a_1}, ..., P_{a_{i-1}}, P_{a_{i+1}}, ..., P_{a_n}) & \text{and} \\ (P_{a_{-i}}, P'_{a_i}) := (P_{a_1}, ..., P_{a_{i-1}}, P'_{a_i}, P_{a_{i+1}}, ..., P_{a_n}) & , \end{array}$$

and let $T \subset A$ be any subset of A. Denote

$$P_T = (P_{a_i})_{a_i \in T}\ , \quad \text{and} \quad P_{-T} = P_{A \backslash T} = (P_{a_i})_{a_i \in A \backslash T}$$

Lemma 16. $\varphi_P = \varphi_{P'} = \varphi_{(P'_{-T},\, P_T)}$ for all subsets $T \subseteq A$.

Proof. Obvious by how P and P' are defined. □

Lemma 17. $\varphi_{P'} = \phi_{P'}$.

Proof. See Appendix A. □

Lemma 18. $\varphi_{(P'_{-T},\, P_T)} = \phi_{(P'_{-T},\, P_T)}$ for any subset $T \subseteq A$.

Proof. See Appendix A. □

Now we are set to prove the theorem:

Theorem 19. A mechanism $\phi : \mathcal{P} \longrightarrow \mathcal{A}$ is individually rational, Pareto-efficient and strategy-proof if, and only if, ϕ is the mechanism of the core (the one selecting the allocation according to Gale's TTC algorithm).

Proof. Let φ be the mechanism in the core.

($\Longrightarrow$) If ϕ is an IR, PE and SP mechanism, then by Lemma 18 taking $T = A$:

$$\left.\begin{array}{l} \varphi_{(P'_{-T},\, P_T)} = \varphi_{(P'_{-A},\, P_A)} = \varphi_P \\ \phi_{(P'_{-T},\, P_T)} = \phi_{(P'_{-A},\, P_A)} = \phi_P \end{array}\right\} \Longrightarrow \quad \varphi = \phi,$$

ϕ is the mechanism of the core.

($\Longleftarrow$) If ϕ is the mechanism of the core for all P (i.e $\phi = \varphi$), it is clear that it satisfies IR and PE (Lemma 1), and from Theorem 13 follows that ϕ will be SP.

□

Remark 5. First, any mechanism that orders the agents and allows that, following this order, the agents receive successively its preferred object among the ones still not chosen by its predecessors (called *dictatorial mechanism in series*), is Pareto-efficient and strategy-proof, but not individually rational; second, the mechanism that always selects the initial endowment ($\psi[A, O, P, \mu]$ $\forall(A, O, P, \mu) \in \mathcal{P}$) is individually rational and strategy-proof, but not Pareto-efficient; third, there exist mechanisms that are individually rational and Pareto-efficient, and at the same time manipulable (so not strategy-proof). In the sense of these three observations, it can be pointed out that individual rationality, Pareto-efficiency and strategy-proofness are mutually independent.

The next example will show the third case proposed above:

Example 6. *Consider the problem* (A, O, P, μ) *where* $|A| = |O| = 4$, $\mu(a_i) = o_i$ *for all* $i = 1, 2, 3, 4$ *and the preference profile is as given in Table 7:*

Table 7: Preferences.

P_{a_1}	P_{a_2}	P_{a_3}	P_{a_4}
o_3	o_4	o_1	o_3
o_2	o_1	o_2	o_2
o_1	o_2	o_3	o_1
o_4	o_3	o_4	o_4

Let φ *be the mechanism in the core associated to Gale's TTC algorithm. Then,*

$$\varphi[A, O, P, \mu] = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ o_3 & o_4 & o_1 & o_2 \end{pmatrix},$$

and define another mechanism ϕ *that coincides with* φ *in every assignment problem except in* (A, O, P, μ) *(i.e.,* $\phi[A, O, Q, \mu] = \varphi[A, O, Q, \mu]$ *for all* $(A, O, Q, \mu) \in \mathcal{P}$ *with* $Q \neq P$*). Select now, for* ϕ *over* (A, O, P, μ)*, the next allocation:*

$$\phi[A, O, P, \mu] = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ o_2 & o_4 & o_1 & o_3 \end{pmatrix}.$$

Since φ *is in the core,* $\varphi[A, O, Q, \mu]$ *is IR and PE for every* $(A, O, Q, \mu) \in \mathcal{P}$*; and therefore,* $\varphi[A, O, Q, \mu]$ *is IR and PE for each* $(A, O, Q, \mu) \in \mathcal{P} \backslash (A, O, P, \mu)$ *(since* $\varphi[A, O, P, \mu] \neq \phi[A, O, P, \mu]$*). But, it is easy to notice that* $\phi[A, O, Q, \mu]$ *is IR:*

$$\phi[A, O, Q, \mu](a) R_a \mu(a) \text{ holds for each agent } a \in A \quad ,$$

and PE considering that there does not exist another allocation in which an agent $a' \in A$ *receives a strictly preferred object than the one assigned by* ϕ *while each agent in* $A \backslash \{a\}$ *receives an object at least as preferred as the one assigned by* ϕ.

Obseve that the only possibility for a' *would be* a_1 *receiving* o_3 *instead of* o_2*; but, in that case, some other agent among the rest would receive an object less desired than the one assigned by* ϕ.

To see that ϕ *can be manipulated, take a profile* $P' = (P_{a_1}, P_{a_2}, P_{a_3}, P'_{a_4})$ *where* $o_1 P'_{a_4} o_2 P'_{a_4} o_3 P'_{a_4} o_4$*. Then,*

$$\phi[A, O, P', \mu] = \varphi[A, O, P', \mu] = \begin{pmatrix} a_1 & a_2 & a_3 & a_4 \\ o_3 & o_4 & o_1 & o_2 \end{pmatrix}.$$

Notice that ϕ *had been chosen such that* $\phi[A, O, Q, \mu] = \varphi[A, O, Q, \mu]$ *for every* $(A, O, Q, \mu) \in \mathcal{P}$ *with* $Q \neq P$.

Hence, as $o_2 P'_{a_4} o_3$ *(i.e.,* $\phi[A, O, P, \mu](a_4) P'_{a_4} \phi[A, O, P', \mu](a_4)$*), agent* a_4 *manipulates mechanism* ϕ *in the assignment problem* (A, O, P', μ) *by reporting* P_{a_4} *instead of the true preference order* P'_{a_4}.

Lastly, another desirable property of the mechanism of the core associated with Gale's TTC algorithm is that the selected allocation corresponds to the one that would be obtained through a competitive market. Alvin E. Roth and Andrew Postlewaite (1977) proved in [36] that, under no indifference between goods, for each assignment problem exists just one allocation that is a competitive equilibrium, and it coincides with the one selected by Gale's TTC algorithm, i.e. with the allocation in the core. It will be demonstrated in the proof of Theorem 20, where theorems 8 and 9, and the concept of competitive allocation will be put together.

Theorem 20. If no trader is indifferent between any goods, then the core defined by weak domination is always non-empty and contains exactly one allocation, which is the unique competitive allocation.

Proof. By Lemma 6 we know that if no trader is indifferent between any goods, a competitive allocation weakly dominates every other allocation, competitive or not. Hence, it is only needed to show that no allocation weakly dominates a competitive allocation.

Let η be a competitive allocation. Associated with η are the coalitions $S_1, S_2, ..., S_K$ (top trading cycles) which are effective for η, and prices π_1, π_2, $..., \pi_K$, respectively; such that $\pi_1 > \pi_2 > ... > \pi_k$. Since η is competitive, there exists a price vector $p = (p_{o_1}, p_{o_2}, ..., p_{o_n})$ such that for each agent a, $p_{\mu(a)} = p_{\eta(a)}$. Every agent in the coalition S_l will have the same price for the object the agent owns (i.e., $p_{o_i} = \pi_l \quad \forall\, a_i \in S_l$).

Take now another allocation ν and suppose the agent weakly dominates η via some coalition T, then:

(i) $\nu(a) \in \mu(T) \quad \forall\, a \in T$,

(ii) $\nu(a) R_a \eta(a) \quad \forall\, a \in T$,

(iii) $\nu(a) P_a \eta(a)$ for some $a \in T$.

Next, let j be the smallest integer such that $S_j \cap T \neq \emptyset$. As η is competitive, if $\nu(a) P_a \eta(a)$ for some $a \in S_j \cap T$, $\nu(a)$ must have been sold at a higher price than $\eta(a)$ ($\pi_k = p_{\nu(a)} > p_{\eta(a)} = \pi_j$). But, this implies that $\nu(a)$ must have been traded in some S_k for $k < j$, and hence, it must have been the initial endowment of an element $b \in S_k$. But, T is effective for ν, therefore $\nu(a) \in \mu(T)$, and so $b \in T$. Consequently, $b \in S_k \cap T$ for $k < j$; which contradicts that j was the smallest integer such that $S_j \cap T \neq \emptyset$.

Therefore, it is false that $\nu(a) P_a \eta(a)$ for each $a \in S_j \cap T$, and since (by hypothesis) there is no indifference, and $\nu(a) R_a \eta(a)$ for all $a \in T$ (by (ii)), it must be that $\nu(a) = \eta(a)$ for all $a \in S_j \cap T$.

Now, assume that $S_1, S_2, ..., S_K$ are simple top trading cycles. Then since $\nu(a) \in \mu(T)\ \forall\ a \in T$ and $S_j \cap T \neq \emptyset$, it follows that $S_j \subseteq T$, and $T \backslash S_j$ is effective for ν (Notice that, $\forall\ a \in S_j$, $\nu(a) = \eta(a)$ and therefore, by how S_j is defined, $\nu(a) \in \mu(S_j)\ \forall a \in S_j$).

Following this way, it will be seen that $\eta(a) = \nu(a)\ \forall a \in T$ (as condition (iii) in the supposition above is not held).

So, ν does not weakly dominate η. Hence, no allocation weakly dominates η. Therefore, η is in the core. Thus, the core is always non-empty and contains exactly one allocation that is the unique competitive allocation. □

In summary, under no indifference allowed, the Top Trading Cycles algorithm from Gale is strategy-proof as a mechanism and selects the only allocation in the core, such allocation is the only competitive allocation. Moreover, it is the only mechanism that is strategy-proof among those individually rational and Pareto-efficient.

Chapter 4

Implementation for the Kidney Paired Donation

Once all the necessary theoretical concepts have been introduced and proved in the previous chapters, the Top Trading Cycles algorithm of Gale can be said to be an ideal mechanism to assign kidneys to patients in a paired kidney exchange program. For the implementation of the algorithm to this context, a computer program will facilitate and streamline the process. The coded computer program consists of different sections with several functions that will make easier to understand and code the main program.

4.1 Input data

The input of the TTC algorithm of Gale is a preference matrix; this is, a profile P. In a real situation, it could be entered into the system by the user, but let us create a random preference matrix deduced from a database derived from real data, in order to perform simulations easily. Both a random database and its associated preference matrix will be generated by a function in the R environment, see [31], taking advantage of the large number of statistical tools it offers.

The random preference matrix is generated in three steps, see the details and codes in Appendix B. Let n be the number of patient-donor couples taking part in the problem. First, a data matrix of dimensions n x 7 is generated by using real data from the ONT retrieved from [21], see Appendix B.1. The rows of the mentioned matrix represent the couples taking part in the program, and the columns collect for each one the necessary data to create a points matrix, in the second stage, according to the selection and prioritization criteria agreed by the ONT in [25], see Appendix B.2. The information collected in the columns for each couple is: the ABO blood groups of the patient and the donor, the matching probability of the patient, the months the patient has spent on dialysis, the ages of the patient and the donor, and the autonomous community where the couple comes from. The points matrix is a square matrix of dimension n x n, where row j represents the donor from couple j and column i the patient from couple i. Hence, in position (j, i) is saved a score that represents how adequate would be for patient i to be assigned the graft from donor j. The scores in the points matrix are determined by characteristics as blood compatibility between the patient and the donor, matching probability of the patient, age difference between the patient and the donor, months the patient has spent on dialysis treatment, geographical location of the couple, and pediatric patients. Finally in the third step, see Appendix B.3, the scores of the points matrix are ordered obtaining an n x n square matrix. In it, column i saves, in descendent order, the donor from whom patient i wants most to receive a kidney, to the donor from whom least.

If we observe Table 8 we can see how the needed time for generating the data, points, and preference matrices, significantly increases while we consider larger values of n.

Table 8: Mean time (sec.) for generating input data.

n	5	10	20	50	100	200	350
Time	0.22	0.24	0.33	0.91	3.23	15.18	62.32

4.2 Implementation

Once the preference matrix is obtained, to code the TTC algorithm, which provides the solution to the assignment problem, C++ programming language is used, see [11]. When coding, two functions will be essential to facilitate the code writing process. The first one, *PreferenceMat*, will read and save the preference matrix created in R environment from a *.dat* file, and give us the number of couples taking part in the problem. The second one, *inCycle*, will let us know if a couple belongs to a cycle in a determined stage of the algorithm. Then, the solving procedure follows the main steps of Algorithm 1 from Chapter 3, see the complete code in Appendix C. The program returns as output the final assignment of kidneys to patients (η), and detailed information about stages, time, transplants, and cycles.

4.3 Simulations and results

In our simulations, we consider 7 dimensions for the number of couples taking part in the problem, $n \in \{5, 10, 20, 50, 100, 200, 350\}$, and 20 random database samples. Then we study two factors: (i) for each dimension we observe the variations in the results, and (*ii*) we compare the results for the different numbers of donor-patient couples. Tables 9 and 10 below show the mean an the coefficient of variation of the results, respectively, where the columns are as follows:

St., number of stages of the TTC algorithm,

Time, computing time in seconds (input time excluded),

Tr./%Tr., number and percentage of compatible transplants in the allocation solution, respectively,

Cy., total number of cycles in the allocation solution,

min./avg./max. Cy/St, minimum, average, and maximum number of cycles per stage, respectively, and

min./avg./max. Len. Cy., minimum, average, and maximum length of a cycle in the algorithm, respectively.

Table 9: Means of the TTC implementation results.

n	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
5	3	0.01	2	38	1	0	0.32	1	2	1.58	2
10	5	0.03	6	55.50	3	0	0.58	1	2	2.09	2
20	8	0.05	13	65	6	0	0.68	1	2	2.39	3
50	16	0.13	36	71.80	15	0	0.91	2	2	2.43	4
100	29	0.28	76	75.50	29	0	1	2	2	2.58	5
200	52	0.76	150	75.20	58	0	1.10	3	2	2.61	6
350	85	1.79	269	76.83	100	0	1.17	4	2	2.71	7

Table 10: Coefficients of variation of the TTC implementation results.

n	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
5	24.18	14.76	74.24	74.24	71.19	0.22*	81.39	59.23	60.94	61.17	62.17
10	19.50	33.86	45.46	45.46	42.05	0.31*	50.72	37.53	23.54	27.52	39.18
20	12.41	16.15	24.45	24.45	26.71	0 *	22.60	21.30	10.91	10.13	17.60
50	10.73	16.94	13.34	13.34	11.72	0 *	8.46	23.54	0	8.97	16.22
100	11.26	22.48	10.11	10.11	11.97	0 *	8.49	22.76	0	6.62	12.56
200	8.08	26.66	7.82	7.82	7.68	0 *	6.45	25.31	0	5.39	21.45
350	6.31	17.31	5.92	5.92	7.03	0 *	4.19	19.66	0	4.03	17.64

*: standard deviation.

(i) **Samples for different databases for a fixed n.** Table 10 shows the coefficients of variation for the results of executing the TTC algorithm with 20 random databases for a fixed number n of patient-donor couples. The variables St., Tr., %Tr., Cy., avg. Cy/St, and avg. Len. Cy. show small variability with respect to the mean for n greater or equal to 50. Moreover, the larger the size of the patient-donor couples set, the smaller the deviation. The minimum variables are quite homogeneous, except for the smallest sizes. Finally, the coefficients of variations performance is not decreasing for the time and maximum variables. The percentage of the deviation from the mean for the time is between 16.15% and 33.86%, for the maximum number of cycles per stage between 19.66% and 59.23%, and for the maximum length of cycles in the algorithm between 12.56% and 62.17%. Except in maybe some particular cases, the maximum number of couples taking part in the same cycle is quite feasible in medical terms even for large number of couples, the largest mean we have obtained for the maximum length of a cycle in a simulation has been 7 for $n = 350$; a positive result on medical grounds considering the problems very long cycles can involve on the operating room coordination, necessary human units, and many other operational aspects. The detailed results of the simulations are collected in Table 16 from Appendix D.

(ii) **Samples for different values of n.** Table 9 shows the mean results, for each number n of patient-donor couples, of the outcomes obtained in Table 16 from Appendix D. It can be appreciated that the larger the size n, the larger the variables, specially the number of stages, time, number of transplants, and number of cycles. Although the time is increasing, it is remarkable the small time needed even for the largest size $n = 350$, less than 2 seconds. The percentage of compatible transplants seems to converge to $3/4$ of n. The minimum number of cycles per stage and the minimum length of a cycle in the algorithm remain constant in 0 and 2 respectively. The average number of cycles per stage and the average length of a cycle in the algorithm are quite similar for each case, around 1 cycle per stage and length 2. The maximum for the previous two variables increases with a speed up of around 1 unit in the different sizes.

When considering the regression analysis of the variables with respect to the size n, astonishing goodness-of-fit have been observed, see the R-squared measures in Table 17 and the expression of the regression curves for the highest determination coefficient in Appendix D.

Chapter 5

Conclusions and further research

The objectives presented in the first chapter of the dissertation have been successfully achieved. The difficulties on the waiting time for kidney transplantation have been identified, and the economic model of trading indivisible objects presented by L. Shapley and H. Scarf has been understood to be able to implement it to the transplants context via the Top Trading Cycles algorithm of Gale. Furthermore, the TTC algorithm has been proved to be the ideal mechanism for trading indivisible goods with no money or other means of exchange. The mechanism is individually rational so that every participating agent obtains an object at least as good as its initial one, a reallocation of the objects cannot give any agent a better object than the one assigned by the mechanism without any other receiving a less preferred one, i.e. it is Pareto-efficient, and no subset of agents can get a better outcome than the allocation given by the algorithm by reallocating their initial objects among them. Besides, it encourages the agents to reveal their true preferences because of being strategy-proof.

The impossibility of adding immunological data to our database, due to the lack of reachable resources about HLA information, leaves a significant gap in the reliability of the simulations. Nevertheless, the main objective of the program has been attained and works properly for the paired kidney exchange assignations. It is successful in completing stages, and satisfies cycles properly, assigning to each patient the best possible graft. The coefficients of variation collected in Table 10 in the previous chapter show us how close are the results of the simulations between each other; this is, the algorithm works in a similar way for problems of the same dimensions, what implies stability and robustness. The outcomes seem to be satisfactory for the available data, with mean bigger than 75% of performable transplants for problem dimensions larger than 50 couples. The mechanism turns out to be a big medical progress in the area of renal donation. Furthermore, the welfare of the patients on the cadaver waiting list would improve due to the relief of the demand on the supply of cadaver kidneys.

Up to this point, the first and most basic form of paired kidney exchange has been discussed. Nevertheless, the research in this topic in the current scientific community is focused in many improvements and variations. An important improvement in the last years has been the possibility of performing the transplants in a cycle in several times thanks to the presence of 'bridge donors'. The 'bridge donor' awaits to donate his kidney while the partner has already been transplanted to enable a bigger number of transplants; Spain introduced this figure for the first time in 2014, see [24], after the 'bridge donor' figure was approved in 2013 by the Transplant Commission of the Interterritorial Health Council, see [10]. Another remarkable advancement is the possibility of carrying out an international paired kidney exchange, which the ONT successfully accomplished for the first time in 2018, and later in 2019; both times between a Spanish patient-donor pair and an Italian one, see [28] and [30]. One of the most relevant variations of the kidney exchange program implementation is the indirect exchange, where one incompatible patient-donor pair exchanges its kidney with the cadaver queue, the patient in the couple receiving high priority on the deceased donors queue, see [37] from Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver (2004). However, this may have a negative impact on type O patients waiting in the cadaver queue. Another version of paired kidney exchange is the altruistic-donor chain, in which the chain starts with an altruistic donor that enters the program willing to donate one of its kidneys, and it ends with a patient-donor couple in the chain donating a kidney to an unpaired recipient on the deceased-donor waiting list, see in [32] an interesting related article by Michael A. Rees et al. (2009). A dynamic kidney exchange model was also presented by M. Utku Ünver (2010) in [43]. These are just some of the possibilities for further research in the paired kidney exchanges that have been studied until the moment this dissertation has been written.

Appendix A

Proofs for auxiliary lemmas

In this appendix the proofs for the auxiliary lemmas for theorems 13 and 19 in Chapter 3 are shown.

Lemma 10. Let $C = (n_1, n_2, ..., n_m)$ be a chain in the graph G_k^P, and $r > k$, then C is a chain in G_r^P if, and only if, n_m is a node of G_r^P.

Proof. First, remember that $(n_1, n_2, ..., n_m)$ is a chain in G_k^P if (n_q, n_{q+1}) is an arc of G_k^P for $q = 1, ..., m-1$. Then, let (n_{m-1}, n_m) be an arc in G_k^P. Notice that the object corresponding to node n_m will be the most preferred one among the available ones at stage k of the algorithm, i.e. in the set $O \backslash (\eta(S_1) \cup \eta(S_2) \cup ... \cup \eta(S_{k-1}))$, for the agent corresponding to node n_{m-1}. Now, as $r > k$, let us prove the two implications :

($\Longrightarrow$) If (n_{m-1}, n_m) is an arc in G_r^P, n_m must be a node in G_r^P.

($\Longleftarrow$) If n_m is a node in G_r^P, since the object corresponding to node n_m will still be the favourite one, among the available objects at stage r of the algorithm, for the agent corresponding to n_{m-1}, then (n_{m-1}, n_m) will still be an arc in G_r^P.

Hence, the proof by induction proceeds forthwith: suppose that n_m is a node in G_r^P, then (n_{m-1}, n_m) is an arc in G_r^P. So, n_{m-1} is a node of G_r^P, and therefore, (n_{m-2}, n_{m-1}) is an arc in this same graph. Thus, (n_{m-2}, n_{m-1}, n_m) is a chain in G_r^P. Following in the same way, if $(n_2, ..., n_m)$ is a chain in G_r^P, since n_2 is a node of such graph, (n_1, n_2) must be an arc, and therefore $(n_1, ..., n_m)$ a chain, in G_r^P. □

Lemma 11. Take P and P' as defined above, and let k and k' be the stages at which agent a_i is removed from the market in TTC_P and $TTC_{P'}$, respectively. Then the graphs G_l^P and $G_l^{P'}$ have the same cycles for $1 \leq l \leq min\{k, k'\} - 1$, and the same nodes for $1 \leq l \leq min\{k, k'\}$.

Proof. Recall that P' differs from P just in the report of an only fixed agent a_i, who declares P'_{a_i} instead of P_{a_i}.

Let N be the set of nodes of G_1^P and $G_1^{P'}$. This two graphs differ only maybe in the arc emanating from node i corresponding to agent a_i, that is the only one who has declared a different preference for profile P'. Both graphs will have the same cycles (and loops) if node i does not belong to a cycle (or loop) in any of them (i.e., if $min\{k, k'\} > 1$). In this case, since $a_i \notin S_1^P$ and $a_i \notin S_1^{P'}$, G_2^P and $G_2^{P'}$ will have the same set of nodes (N without the nodes corresponding to agents in $S_1^P = S_1^{P'}$). As argued before, both graphs will differ only maybe in the arc emanating from i, and will have the same cycles (and loops) if $min\{k, k'\} > 2$. The lemma follows by induction: suppose that in stage l of TTC_P and $TTC_{P'}$ the graphs G_l^P and $G_l^{P'}$ have the same nodes and the same cycles (and loops) (i.e., node i will not belong to a cycle (or loop) and will continue unallocated). Then in stage $l+1$, both G_l^P and $G_l^{P'}$ will have the same nodes. They will only have the same cycles (and loops) if node i is not in a cycle (or loop) (i.e., if $min\{k, k'\} - 1 \geq l + 1$).

Therefore, G_1^P and $G_1^{P'}$ will have the same set of nodes until the stage in which a_i is removed from the problem in TTC_P or $TTC_{P'}$, thus, for $1 \leq l \leq min\{k, k'\}$. For the same reason, G_l^P and $G_l^{P'}$ will have the same cycles and loops until exactly one stage before a_i is removed from one of the problems (i.e., the stage in which a_i belongs to a cycle or loop in one of the problems); hence, for $1 \leq l \leq min\{k, k'\} - 1$. □

Lemma 12. Let P'' be a preference profile which differs from P' only in the report of agent a_i, where P''_{a_i} is any preference such that $\eta'(a_i)P''_{a_i}\eta'(a_q)$ for all $a_q \neq a_i$ (being η' the allocation resulting from $TTC_{P'}$). Then if η'' is the resulting allocation from $TTC_{P''}$, $\eta''(a_i) = \eta'(a_i)$.

Proof. Let k' and k'' be the stages at which a_i is removed from the market in $TTC_{P'}$ and $TTC_{P''}$ respectively, and let $\eta'(a_i) = \mu(a_j) = o_j$ for some $a_j \in A$.

Suppose a_i is still on the market at period k' of $TTC_{P''}$ (i.e., $k' \leq k''$), then by Lemma 11 the graphs $G_{k'}^{P'}$ and $G_{k'}^{P''}$ will have the same nodes, and moreover, $G_{k'-1}^{P'}$ and $G_{k'-1}^{P''}$ the same cycles. But, as $\eta'(a_i) = \mu(a_j)$ the arc (i, j) is in $G_{k'}^{P'}$, and besides, it is in $G_{k'}^{P''}$ since P''_{a_i} ranks $\mu(a_j)$ first for agent a_i over all the objects (i.e., the arc (i, j) is in the graph of every stage in which j is still a node). Therefore, $G_{k'}^{P'} = G_{k'}^{P''}$ considering that the only aspect in which $G_{k'}^{P'}$ and $G_{k'}^{P''}$ could differ is in the arc emanating from i, that it has been seen to be the same. In this case, i belongs in $G_{k'}^{P''}$ to the same cycle as in $G_{k'}^{P'}$. Therefore, $\eta'(a_i)$ is allocated to a_i in step k' of $TTC_{P''}$, and hence $\eta''(a_i) = \eta'(a_i)$. Lemmas 10 and 11 also imply that a_i (and/or node

i) can not be removed from the problem later than $\eta''(a_i)$ (and/or node j) in $TTC_{P''}$, then must be $k'' \leq k'$ too. □

Lemma 14. Let α, ω be Pareto-efficient with respect to profile P, and suppose $\alpha \neq \omega$. Then $J(\alpha, \omega, P)$ is not empty.

Proof. By reductio ad absurdum, suppose that:

$$J(\alpha, \omega, P) = \{a \in A : \alpha(a) P_a \omega(a)\} = \emptyset \implies \nexists\, a \in A \text{ s.t. } \alpha(a) P_a \omega(a) \quad .$$

Therefore, $\omega(a) R_a \alpha(a)\ \forall\, a \in A$. Hence, there are two possibilities:

(i) $\omega(a) P_a \alpha(a)$ for some $a \in A \implies \alpha$ is not PE w.r.t. P,

(ii) $\nexists\, a \in A$ s.t. $\omega(a) P_a \alpha(a) \quad \implies \quad \omega = \alpha$,

each of them leading to a contradiction. □

Lemma 15. Let η be the allocation in the core of assignment problem (A, O, P, μ), and an allocation ν individually rational and Pareto-efficient w.r.t. profile P, being $\eta \neq \nu$. Then exists $a \in J(\eta, \nu, P)$ such that

$$\eta(a) P_a \nu(a) P_a \mu(a).$$

Proof. First, notice that:

· η is the allocation in the core. Hence, η is IR and PE by Lemma 1.

· ν is IR and PE by hypothesis in Lemma 15.

By Lemma 14, since both η and ν are PE w.r.t. P and $\eta \neq \nu$, then:

$$J(\eta, \nu, P) \neq \emptyset \implies \exists\, a \in A \text{ s.t. } \eta(a) P_a \nu(a)\ . \tag{A.1}$$

By reductio ad absurdum, suppose now that:

$$\nexists\, a \in J(\eta, \nu, P) \text{ s.t. } \eta(a) P_a \nu(a) P_a \mu(a)\ . \tag{A.2}$$

By (A.1) and (A.2), and since $\eta(a) P_a \nu(a)$:

$$\nexists\, a \in J(\eta, \nu, P) \text{ s.t. } \nu(a) P_a \mu(a)\ . \tag{A.3}$$

And, ν is IR $\iff \nu(a) R_a \mu(a)\ \forall a \in A \overset{(A.3)}{\implies} \nu(a) = \mu(a)\ \forall\, a \in J(\eta, \nu, P)$.

Now, let be $T = A \backslash J(\eta, \nu, P)$. The restriction of allocation ν to the coalition T, ν^T, is a T-allocation, i.e. $\nu^T(a) \in \mu(T)\ \ \forall\, a \in T$. Additionally, considering that, by Lemma 14, $J(\nu,\eta,P) \neq \emptyset$ (and the disjointness of $J(\eta,\nu,P)$ and $J(\nu,\eta,P)$):

$$\exists\, a \in T \;\text{ s.t. }\; \nu(a) P_a \eta(a)\ , \tag{A.4}$$

and for how T is defined:

$$\nu(a) R_a \eta(a)\ \forall\, a \in T\ . \tag{A.5}$$

Hence, by (A.4), (A.5) and the effectiveness of ν^T, ν^T weakly dominates η. Therefore, η does not belong to the core, which is a contradiction. □

Lemma 17. $\varphi_{P'} = \phi_{P'}$.

Proof. Notice that Lemma 15 is applicable because $\varphi_{P'}$ is the mechanism in the core of the assignment problem, $\phi_{P'}$ is IR and PE for any profile P'. By reductio ad absurdum, suppose that $\varphi_{P'} \neq \phi_{P'}$, then:

$$\exists\, a \in J(\varphi_{P'}, \phi_{P'}, P') \;\text{ s.t. }\; \varphi_{P'}(a)\, P'_a\, \phi_{P'}(a)\, P'_a\, \mu(a). \tag{A.6}$$

But, by how profile P' is defined above, and by the fact that $\varphi_P = \varphi_{P'}$ from Lemma 16, for each $a \in A$, there are two cases:

(1) If $a \in T_P$, in preference P' $\varphi_P(a)$ (and therefore $\varphi_{P'}(a)$) is immediately followed by the object $\mu(a)$.

(2) Otherwise, if $a \in A \backslash T_P$, then $P'_a = P_a$ and $\nexists\, o \in O$ s.t. $\varphi_{P'}(a) P'_a o P'_a \mu(a)$ (remember that φ is IR), there are two possibilities:

- · $\varphi_{P'}(a)$ is immediately followed by $\mu(a)$, or
- · $\varphi_{P'}(a) = \mu(a)$ in P'_a

In any case, there would not exist such object $\phi_{P'}(a)$ in (A.6), which is a contradiction. □

Lemma 18. $\varphi_{(P'_{-T},\, P_T)} = \phi_{(P'_{-T},\, P_T)}$ for any subset $T \subseteq A$.

Proof. By Lemma 17, $\varphi_{P'} = \phi_{P'}$. Let us consider two cases:

(1) $T \subseteq A \backslash T_P$, and

(2) $T \subseteq T_P$

First, take a subset $T \subseteq A \backslash T_P$, then $\forall\, a \in T$, $P'_a = P_a$, hence:

$$\varphi_{(P'_{-T},\, P_T)} = \varphi_{(P'_{-T},\, P'_T)} = \varphi_{P'} = \phi_{P'} = \phi_{(P'_{-T},\, P'_T)} = \phi_{(P'_{-T},\, P_T)}.$$

Therefore, it is just needed to prove Lemma 18 for $T \subseteq T_P$, i.e. the second case. By induction on the size of subset T:

· If $|T| = 0$, by Lemma 17:

$$\varphi_{(P'_{-T},\, P_T)} = \varphi_{(P'_{-\emptyset},\, P_\emptyset)} = \varphi_{P'} = \phi_{P'} = \phi_{(P'_{-\emptyset},\, P_\emptyset)} = \phi_{(P'_{-T},\, P_T)},$$

and Lemma 18 is satisfied.

· Assume by induction hypothesis that Lemma 18 holds for any $|T| \leq k$.

· Let us prove for $|T| = k+1$. By reductio ad absurdum, suppose $\varphi_{(P'_{-T},\, P_T)} \neq \phi_{(P'_{-T},\, P_T)}$. Denote $Q = (P'_{-T},\ P_T)$. Then by Lemma 15 (which can be applied as φ is the mechanism of the core and ϕ is IR and PE):

$$\exists\ a \in J(\varphi_Q, \phi_Q, Q) \quad \text{s.t.} \quad \varphi_Q(a)\ Q_a\ \phi_Q(a)\ Q_a\ \mu(a) \tag{A.7}$$

If $a \in A \backslash T$ then, by Lemma 16, $\varphi_P = \varphi_{P'} = \varphi_Q$, i.e. $\varphi_Q(a) = \varphi_P(a)$, and from (A.7):

$$\varphi_P(a)\ P'_a\ \phi_Q(a)\ P'_a\ \mu(a), \tag{A.8}$$

noticing that since $Q = (P'_{-T},\ P_T)$, $Q_a = P'_a$ for each $a \in A \backslash T$. It is impossible by the construction since in P'_a either $\mu(a)$ follows $\varphi_P(a)$ immediately, or $\varphi_P(a) = \mu(a)$, for all $a \in A \backslash T$.

Therefore such a should be in T. Then by Lemma 16 we have:

$$\varphi_Q(a) = \varphi_{P'}(a) = \varphi_{(Q_{-a},\, P'_a)}(a), \tag{A.9}$$

Utilizing that $(Q_{-a}, P'_a)_a = P'_a$, and hence, $\varphi_{P'}(a) = \varphi_{(Q_{-a},\, P'_a)}(a)$. And by the induction hypothesis:

$$\varphi_{(Q_{-a},\, P'_a)}(a) = \phi_{(Q_{-a},\, P'_a)}(a). \tag{A.10}$$

By induction hypothesis, since $|T \backslash \{a\}| = k$ it can be assumed that:

$$\varphi_{(P'_{-T\backslash\{a\}},\, P_{T\backslash\{a\}})} = \phi_{(P'_{-T\backslash\{a\}},\, P_{T\backslash\{a\}})};$$

that is, $\varphi_{P'}(a) = \phi_{P'}(a)$, and therefore, (A.10) holds. Hence, from (A.7), (A.9), and (A.10):

$$\phi_{(Q_{-a},\, P'_a)}(a)\ P_a\ \phi_Q(a). \tag{A.11}$$

Observe that by how Q is defined, $Q_a = P_a$ and by (A.9) and (A.10) $\varphi_Q(a) = \phi_{(Q_{-a},\, P'_a)}(a)$.

Substituting for Q, from (A.11):

$$\phi_{((P'_{-T},\, P_T)_{-a},\, P'_a)}(a)\ P_a\ \phi_{(P'_{-T},\, P_T)}(a)$$

and a would be receiving a better object by reporting preference P'_a instead of the truthful one P_a, which contradicts that the mechanism ϕ is strategy-proof. Therefore, there does not exist such agent a (contradiction). Lemma

18 is satisfied also for $|T| = k + 1$.

Hence, the lemma is true for any $T \subseteq A$. $\square$

Appendix B

Data, points, and preference matrices

The implementation of the Top Trading Cycles algorithm of Gale for paired kidney donation will be carried out by a computer program coded in C++ language. The purpose is to combine the patients and donors of n incompatible patient-donor pairs in the most profitable way and using the criteria of selection and prioritization agreed by the ONT in 2015 for the Spanish paired renal donation program, see [25]. Some simplifications when applying the criteria will be considered due to the limited time, and shortage of information or difficulties to obtain it; they will be discussed later.

As mentioned in Chapter 4, R environment has been utilized to generate both the data matrix and its corresponding preference matrix. The appendix will be organized as follows: first, a data matrix is generated based on the data collected by the ONT; then, a points matrix is completed utilizing the previous matrix and the selection and prioritization criteria proposed by the ONT; lastly, the scores are ordered to obtain the preference matrix.

B.1 Generation of a data matrix

A realistic data matrix will be generated using the most influential information when assigning pairs. All the necessary data is saved in a matrix of n rows and exactly seven columns, being n the number of couples taking part in the program. The first, third, forth, and fifth columns collect information about the blood group, immunology, dialysis stage, and age of the patients, respectively; the second and sixth columns gather information about the donor's blood group and age, respectively; and the last one details the autonomous community where both the patient and the related initial donor come from. Let us explain first in subsection B.1.1 the real database and information we will use for the simulations, then in subsection B.1.2 a detailed explanation about how the data in the matrix is generated will be given.

B.1.1 National registry of patient-donor pairs

The Spanish Paired Renal Donation Program developed in 2009, possesses a national registry of patient-donor pairs available, created in the ONT. The registry meets the requirements of the Organic Law 15/1999, of December 13, of Personal Data Protection, see [13], and its functions are:

- Gather the necessary information for the clinical and immune assessment of the pairs.
- Identify the possible combinations of pairs for an exchange by compatibility criteria, and apply prioritization criteria in the cases of pairs with more than one exchange choice.
- Provide each member of the pair a unique identifying number with privacy purposes.

The registry collects the following variables of the agents and donors:

(i) **Identity data (donor and patient):** name, hospital, medical history number, ID card, program enter date.

(ii) **Demographic data (donor and patient):** age (birth date), gender, patient-donor relationship.

(iii) **Blood group (donor and patient)**

(iv) **Immunological data:**

- **HLA Typing (donor and patient):** generic HLA A, B, C, DRB1, and DQB1.
- **Rate of cytotoxic antibodies (patient):** Panel Reactive Antibodies (PRA) estimated for classes I and II in the last year.
- **Prohibited HLA specificities, determination date and used technique.**

(v) **Clinical data (patient):** dialysis state and accumulated months in dialysis (including all the treatment periods).

(vi) **Cause of inclusion on the program (donor respect to patient):** blood type incompatibilities, positive crossmatch, other cases.

(vii) **Dialysis starting date.**

(viii) **Results of crossmatch testing executed with pairs from the program.**

In renal transplantation, a positive crossmatch between donor cells and recipient serum is related to early rejection or graft loss; therefore, a positive result in crossmatch testing could in some cases be considered a contradiction to renal transplantation, see [7].

A patient can enter the patient-donor pair registry with more than one intended initial donor, and compatible pairs willing to be included in the program (to obtain better graft compatibility, a donor more close in age, or because of several other reasons) can participate. But, for this dissertation, let us consider a database in which each patient enters the registry together with a unique incompatible donor. Rename the national registry of patient-donor pairs as Patients with Incompatible Donor registry (PID registry) for a more clear understanding.

To generate a data matrix as close as possible to reality, let us use the information from the PID registry from 2009 to 2014 in Spain, see [21], which contains 316 patients with a single incompatible initial donor. Due to the large amount of immunological groups, the difficulty in finding such type of immunological data, and given the time constraints, we will not to use the HLA typing characteristics; this will involve some restrictions and decisions discussed later.

B.1.2 Data matrix

In the data matrix there are as many rows as patient-donor pairs (n) are included in the assignment problem, and exactly seven columns; that is, the factors related to the members in the patient-donor couples that will be useful later for filling the points matrix. Let us start with a zero matrix of dimensions nx7 as shown in *CodeC0.R*.

CodeC0.R

```
# Data matrix, points matrix, and preference matrix (input data sets
    for Gale's TTC algorithm implementation).
# Miren Lur Barquin Torre. 19 February 2020.
library(truncnorm)
library(MASS)
#zero matrix (nCouples x 7)
nCouples<-10
dataMat<-matrix(data=0,nCouples,7,byrow=TRUE)
dataMat<-as.data.frame(dataMat)
```

Then the information will be organized as follows.

The **first** and **second** columns of the matrix will save the blood groups of the patients and donors respectively. That way, row i will have in first position agent a_i's blood group, and in second a_i's initial donor's blood type.

Tables 11 and 12 show useful information to fill the first two columns of the matrix. The first one contains the percentages of agents and donors that belong to each blood group in the PID registry between 2009 and 2014, and the patient-donor blood group compatibilities. The blood group depends on two proteins A and B that can be present or not in each person's blood; this will determine the blood group to which the individual belongs. If not protein A nor protein B are in the blood, the blood type is O, if both A and B are present, the blood type is AB, and if either just protein A or just protein B is present in the blood, the blood type will be A and B, respectively. A patient cannot receive, from a donor, a protein that he or she does not have; hence, the patient-donor compatibilities are as shown in Table 11.

Table 11: Blood type percentages PID 2009-2014 and patient-donor blood compatibilities.

Donors (%) / Patients (%)	O (29.7%)	A (49.9%)	B (15.8%)	AB (4.6%)
O (55.9%)	✓			
A (28.4%)	✓	✓		
B (13 %)	✓		✓	
AB (2.7%)	✓	✓	✓	✓

Notice that we need to work with conditioned probabilities when deciding to what blood group does a donor belong, considering its initial partner's blood type and if they are compatible or not.

Example 7. *Let a patient have blood type B, and be ABO incompatible with its initial donor. We want to determine to which blood group the initial donor belongs. By the hypothesis, the donor should belong to the blood group A or AB, which have probabilities of* 0.499 *and* 0.046*, respectively. Hence, the probability of the donor belonging to the blood group A is* $0.499/(0.499 + 0.046)$*, and to the blood group B:* $0.046/(0.499 + 0.046)$.

Table 12 shows the percentage of patient-donor couples that entered the program because of blood group incompatibilities (ABOi) between 2009 and 2014. The remaining incompatible couples entered the program because of the presence of antibodies in the patient against the donor's HLA (HLAi).

Table 12: Reasons for inclusion in PID registry 2009-2014.

REASON FOR INCLUSION	(%)
ABOi	50.7
HLAi	49.3

For the first column, it is enough to create a random set of n elements that follow a multinomial distribution according to the probabilities deduced from the percentages given by the 'Patients (%)' column in Table 11 for each ABO blood type, see *CodeC1.R*.

CodeC1.R

```
# C1: Patients' blood group.

probABOPatient <- c(0.559,0.284,0.13,0.027)
auxABOp<-rmultinom(1, nCouples , probABOPatient )

dataMat[,1]=rep(c("O","A","B","AB"),auxABOp)
```

In the second column, we need to consider each couple's reason for inclusion in the program, in addition to the percentages of the donors' blood type distribution given in Table 11. The 50.7% of the couples entering the program were ABO incompatible. To fill the column, for each donor we randomly decide, first, if it is ABO compatible with its partner, and next which blood group the initial donor belongs to. In order of these two decisions to follow every required condition, let us generate for each of them a random number following a uniform distribution in $[0, 1]$, and utilize the probabilities derived from tables 11 and 12. See *CodeC2.R*.

CodeC2.R

```
# C2: Donors' blood group.

o=0.297; a=0.499; b=0.158; ab=0.046 #Prob. ABO group donors.

p<-0.507    # The 50.7% of initial patient-donor pairs have ABO
    incopatibilities.
for(i in 1:nCouples){
  t<-runif(1,min=0, max=1)
  compat<-runif(1,min=0, max=1)
  if((compat <= p) & (dataMat[i,1]!="AB")){ # if incompatible (group
     AB has no ABO incompatibilities)
    if(dataMat[i,1]=="O"){        # O group patient
      if(t<=(a/(a+b+ab))){
        dataMat[i,2]="A"}
```

```
    else if(t<=((a+b)/(a+b+ab))){
      dataMat[i,2]="B"}
    else{dataMat[i,2]="AB"}}

  else if(dataMat[i,1]=="A"){    # A group patient
    if(t<=(b/(b+ab))){
      dataMat[i,2]="B"}
    else{dataMat[i,2]="AB"}}

  else if(dataMat[i,1]=="B"){    # B group patient
    if(t<=(a/(a+ab))){
      dataMat[i,2]="A"}
    else{dataMat[i,2]="AB"}}

}else{  # if compatible
  if(dataMat[i,1]=="O"){         # O group patient
    dataMat[i,2]="O"}

  else if(dataMat[i,1]=="A"){    # A group patient
    if(t<=(o/(o+a))){
      dataMat[i,2]="O"}
    else{
      dataMat[i,2]="A"}}

  else if(dataMat[i,1]=="B"){    # B group patient
    if(t<=(o/(o+b))){
      dataMat[i,2]="O"}
    else{dataMat[i,2]="B"}}

  else{                         # AB group patient
    if(t<=o){
      dataMat[i,2]="O"}
    else if(t<=(o+a)){
      dataMat[i,2]="A"}
    else if(t<=(o+a+b)){
      dataMat[i,2]="B"}
    else{dataMat[i,2]="AB"}}}}
```

The **third** column saves the matching probability (MP) of each patient. This factor determines which patient had the least chance of finding a compatible donor in the registry. The MP is calculated for every patient in every match, and it satisfies the following formula developed by Keizer et al. (2005), see [15], where PRA are the Panel Reactive Antibodies:

$$MP = (100 - \text{PRA}) \frac{\#\text{ ABO compatible donors without unacceptable HLA specificities}}{\#\text{ ABO compatible donors}};$$

But due to the difficulty that establishing immunological incompatibilities entails, and the large number of immunological groups as mentioned before, it will be considered that every pair has the same immunological compatibility. Because of the previous simplification, the number of compatible donors will be the same as the number of compatible donors with unacceptable HLA specificities, and therefore the MP will be considered as 100%−PRA%. According to the PID registry, the PRAs among the patients are distributed as follows:

Table 13: PRA PID 2009-2014.

PRA (%)	PATIENTS (%)
0	46.45
$[1, 50)$	14.2
$[50, 80)$	10.06
$[80, 100]$	29.29

When coding, a uniform distribution in the interval [0,1] will be useful to determine to which PRA category belongs each patient according to Table 13. On the other hand, let us obtain a random PRA in the corresponding interval by using a uniform distribution. The code for this column is *CodeC3.R*.

CodeC3.R

```
# C3: 'matching' probability.

for(i in 1:nCouples){
  t=runif(1,min=0, max=1)
  if(t<=0.4645){
    dataMat[i,3]=100-0}
  else if(t<=(0.142+0.4645)){
    dataMat[i,3]=100-(runif(1,min=1, max=49.99))}
  else if(t<=(0.1006 +0.142+0.4645)){
    dataMat[i,3]=100-(runif(1,min=50, max=79.99))}
  else{
    dataMat[i,3]=100-(runif(1,min=80, max=100.99))}}
```

The months that each patient has spent on dialysis treatment are collected in column **four**. This time, the PID registry presents the variable with the mean and standard deviation in a normal distribution, and the minimum and maximum values. The mean of the dialysis for the patients is 55.7 months, and the standard deviation 61.7 months. Patients in the utilized database have been from a minimum of 0 to a maximum of 297 months on dialysis. To obtain the values of this characteristic for the patients in the data matrix we have considered random values in a truncated normal distribution. Nevertheless, there is a known percentage of patients that have not started dialysis yet. Therefore, mind the 21.8% of the patients that have been 0 months in dialysis according to Table 14.

Table 14: Dialysis state PID 2009-2014.

DIALYSIS STATE	PATIENTS (%)
Peritoneal dialysis	13.3
Hemodialysis	59.1
Predialysis	21.8
Dysfunctional Transplant	5.8

The code to obtain the fourth column is *CodeC4.R*.

CodeC4.R

```
# C4: number of months in dialisis.

auxCol4<-replicate(nCouples,0)
auxMonths<-rtruncnorm(round(nCouples*0.782), a=0, b=297, mean=55.7,
    sd=61.7)

auxCol4[1:round(nCouples*0.782)]=round(auxMonths)  # 21.8% of the
    patient have not started dialysis yet.

auxCol4=sample(auxCol4)
dataMat[,4]=auxCol4
```

The **fifth** and **sixth** columns, contain the ages of each patient and each donor, respectively. For patients' ages, according to the 2009-2014 PID registry, we have a normal distribution of mean 47.2 years and standard deviation 11.9 years. The minimum age of patients taking part in the program is 7 and the maximum age is 72. Random values in a truncated normal distribution are selected to obtain the needed data, as shown in *CodeC5.R*.

CodeC5.R

```
# C5: Patients' ages.

auxAgeP<-rtruncnorm(nCouples, a=7, b=72, mean=47.2, sd=11.9)
dataMat[,5]=round(auxAgeP)
```

In the sixth column, initial donors' ages are obtained in the same way as patients' ages in column five, this time with mean of 49.6 years, standard deviation of 10.6 years, and minimum and maximum values of 19 and 74 years, respectively, see *CodeC6.R*.

CodeC6.R

```
# C6: Agents' ages.

auxAgeD<-rtruncnorm(nCouples, a=19, b=74, mean=49.6, sd=10.6)
```

```
dataMat[,6]=round(auxAgeD)
```

Finally, the **last** column collects information about the autonomous community from which each individual comes from. To facilitate the analysis, consider that the patient and donor in the same couple come from the same autonomous community. The data is distributed as shown in Table 15.

Table 15: Autonomous Community PID 2009-2014.

AUTONOMOUS COMMUNITY	(%)
Andalucia	31.7
Aragon	1.2
Canary Islands	1.5
Cantabria	0.9
Catalonia	45
Valencian Community	1.5
Galicia	5.6
Madrid	5
Basque Country	7.7

The column has been obtained randomly respecting the percentages given in Table 15, working the same way as in the first column, as shown in *CodeC7.R*.

CodeC7.R

```
# C7: Autonomous Communities.
# A.C.            Notation     %
# Andalucia          1      31.7
# Aragon             2       1.2
# Canary Islands     3       1.5
# Cantabria          4       0.9
# Catalonia          5      45
# Valencian C.       6       1.5
# Galicia            7       5.6
# Madrid             8       5
# Basque Country     9       7.7

probAutComm<-c(0.317,0.012,0.015,0.009,0.45,0.015,0.056,0.05,0.077)
auxAutComm<-rmultinom(1, nCouples , probAutComm )
auxCol7<-replicate(nCouples,0)

auxCol7=rep(c(1,2,3,4,5,6,7,8,9),auxAutComm)

auxCol7=sample(auxCol7)

dataMat[,7]=auxCol7
```

Example 8 presents a data matrix generated for 10 patient-donor couples.

Example 8.

$$dataMat = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \end{array} \begin{array}{c} \begin{array}{ccccccc} C1 & C2 & C3 & C4 & C5 & C6 & C7 \end{array} \\ \begin{pmatrix} O & O & 100.00 & 28 & 42 & 43 & 5 \\ O & O & 100.00 & 63 & 41 & 52 & 1 \\ O & O & 11.79 & 0 & 47 & 41 & 5 \\ O & O & 100.00 & 87 & 50 & 52 & 9 \\ A & A & 8.65 & 79 & 34 & 41 & 1 \\ A & B & 100.00 & 93 & 57 & 55 & 5 \\ A & AB & 100.00 & 42 & 53 & 60 & 1 \\ A & A & 31.61 & 0 & 48 & 52 & 1 \\ B & A & 100.00 & 99 & 29 & 49 & 5 \\ B & O & 61.55 & 39 & 30 & 47 & 1 \end{pmatrix} \end{array}$$

B.2 Generation of the points matrix

For the generation of the preference matrix for the agents in the data matrix obtained in the previous section, let us first utilize a points system agreed by the ONT, see [25], for defining the preferences between grafts. In this stage, let us start with a zero square matrix of dimension n in which the columns will represent the patients and the rows the donors. Next, let us add to each position (j, i) the corresponding points in consideration of the characteristics of patient i and donor j, both as individuals and related between them.

An explanation on the selection and prioritization criteria that will be utilised is given in B.2.1, previously to filling the points matrix in subsection B.2.2.

B.2.1 Selection and prioritization criteria

The purpose is to recognise for each patient the most suitable grafts in the program and to consider those cases in which the recipient is less expected to obtain a transplant because of medical restrictions.

On the one hand, the selection criteria will be evaluated, and on the other, the prioritization criteria.

Selection criteria:

1. Blood group compatibility.

2. Immunological criteria.

Prioritization criteria:
After considering the selection criteria, if there are more than one compatible donor for a patient, some immunological and not immunological priorities will be established:

1. **Blood group compatibility:** if the patient an the potential donor are isogroup (+30 points (p.)).
2. **MP:**

   0-25%: +30 p.

   25-50%: +20 p.

   50-75%: +10 p.

   75-100%: +0 p.

3. **Patient-donor age difference:** priority if difference $\leq$ 10 years:

   - If the difference is $\leq$ 10 years in both couples: +30 p.
   - If the difference is $\leq$ 10 years in just one couple: +15 p.
   - If the difference is $>$ 10 years in both couples: +0 p.

   In the case of pediatric patients or patients with PRA greater than 50%, blood group O, or incompatible donor of group AB, age difference of more than 10 years will be accepted and they will receive 30 points (+30 p.).

4. **Time on dialysis:** priority to the couple that has more accumulated time on dialysis: 0.05*(months in dialysis) p.
5. **Pediatric patients ($\leq$ 16 years old):** (just in the case of ties in the score) the donor must be less or equal to 50 years old and will obtain extra 30 points (+30 p.).
6. **Geographical location:** if the potential donor and the patient are from the same autonomous community +5 p.

B.2.2 The points matrix

Let us create a zero square matrix with n columns and rows, being n the number of couples taking part in the program. The columns will represent the patients and the rows the participating donors. To complete the matrix, follow four steps:

1. Each patient would rather to stay with its initial incompatible donor, than to obtain an incompatible graft from some other donor. To represent such incompatibility and preference, let us give -1 points to every element in the diagonal of the matrix.

2. Blood incompatibilities between the patient i and the potential donor j are not accepted, hence element (j, i) in the matrix will receive -2 points.

3. The points determined by the restrictions in subsection B.2.1 will be added to each element of the matrix that remains with 0 points.

4. If in any column there are ties between scores, the points relative to autonomous communities will be added where corresponds.

The code is shown in *PointsMat.R*.

PointsMat.R

```
# Zero matrix (nCouples x nCouples). Rows->donors & columns->
    patients.

pointsMat<-matrix(data=0,nCouples,nCouples,byrow=TRUE)

# First: all the patient-donor pairs that enter the program are
    incompatible, therefore we do not need to add points to position
     (i,i) for all i=1,..., nCouples but also is known that for any
    patient is better to stay with its initial object than receive
    another incompatible one. To represent this and make easier to
    order the preferences, let's give elements (i,i) in the diagonal
     of the matrix value -1 and incompatibilities -2 (s.t. they stay
     under the initial object in the preferences).

for(i in 1:nCouples){
  pointsMat[i,i]=-1
}

for(i in 1:nCouples){
  for(j in 1:nCouples){
    if(i!=j){ # non isogroup
      if((dataMat[i,1]=="O" & dataMat[j,2]!="O")|(dataMat[i,1]=="A"
    &(dataMat[j,2]=="B"|dataMat[j,2]=="AB"))|(dataMat[i,1]=="B" & (
    dataMat[j,2]=="A"|dataMat[j,2]=="AB"))){
        pointsMat[j,i]=-2
      }else{
        # ISOGROUP
        if(dataMat[i,1]==dataMat[j,2])
          {pointsMat[j,i]=pointsMat[j,i]+30}
        # MATCHING PROBABILITY
        if(dataMat[j,3]<=25){pointsMat[j,i]=pointsMat[j,i]+30}
        else if(dataMat[j,3]<=50){pointsMat[j,i]=pointsMat[j,i]+20}
        else if(dataMat[j,3]<=75){pointsMat[j,i]=pointsMat[j,i]+10}
        # AGE DIFFERENCE
        # In the case of sensitized recipient (PRA>50), ABO group O
    recipients, the donor belongs to ABO group AB or if the
```

```
    recipient is a child, they will accept the age difference > 10
    and they will receive 30 points.
        if( dataMat[j,5]<16|dataMat[j,1]==1|dataMat[j,2]==4|dataMat[
    j,3]<=50){pointsMat[j,i]=pointsMat[j,i]+30}
        else if(abs(dataMat[i,5]-dataMat[j,6])<=10 & abs(dataMat[j
    ,5]-dataMat[i,6])<=10){pointsMat[j,i]=pointsMat[j,i]+30}
        else if (abs(dataMat[i,5]-dataMat[j,6])<=10 | abs(dataMat[j
    ,5]-dataMat[i,6])<=10){pointsMat[j,i]=pointsMat[j,i]+15}
        # MONTHS ON DIALYSIS
        pointsMat[j,i]=pointsMat[j,i]+0.05*dataMat[j,4]
        # Child recipients (<16 years) will receive +30 points (
    already done in AGE section) and the donor must be <=50 years
    old.
        if(dataMat[i,5]<16&dataMat[j,6]>50){pointsMat[j,i]=-2}
        # The Autonomous Community will be used later just in the
    case of tie.
      }}}}

# in the case that there are ties in the columns, if both pairs
    belong to the same Autonomous community add 5 points.
for(i in 1:nCouples){
  repetitions<-c()
  duplicates=pointsMat[,1][duplicated(pointsMat[,1])]
  for(k in duplicates){
    if(is.element(k, repetitions)==FALSE & k!=-2){
      repetitions<-append(repetitions,k)
      for(j in 1:nCouples){
        if(pointsMat[j,i]==k){
          if(dataMat[i,7]==dataMat[j,7]){
            pointsMat[j,i]=pointsMat[j,i]+5
          }}}}}}
```

By using the data matrix obtained in Example 8, the points matrix in Example 9 is obtained.

Example 9. *The points matrix (PointsMat) related to the data matrix in Example 8 is:*

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
o_1	−1.00	61.40	61.40	61.40	31.40	1.40	16.40	31.40	16.40	16.40
o_2	63.15	−1.00	63.15	48.15	18.15	18.15	18.15	18.15	18.15	18.15
o_3	90.00	90.00	−1.00	90.00	60.00	60.00	60.00	60.00	60.00	60.00
o_4	64.35	49.35	64.35	−1.00	19.35	34.35	34.35	34.35	19.35	19.35
o_5	−2.00	−2.00	−2.00	−2.00	−1.00	93.95	93.95	93.95	−2.00	−2.00
o_6	−2.00	−2.00	−2.00	−2.00	−2.00	−1.00	−2.00	−2.00	49.65	49.65
o_7	−2.00	−2.00	−2.00	−2.00	−2.00	−2.00	−1.00	−2.00	−2.00	−2.00
o_8	−2.00	−2.00	−2.00	−2.00	80.00	80.00	80.00	−1.00	−2.00	−2.00
o_9	−2.00	−2.00	−2.00	−2.00	34.95	49.95	49.95	49.95	−1.00	−2.00
o_{10}	56.95	56.95	56.95	56.95	11.95	26.95	26.95	26.95	11.95	−1.00

B.3 The preference matrix

The preference matrix is a square matrix of dimension n x n, being n the number of couples taking part in the problem. Once we have the points matrix, let us order the scores of each column decreasingly, this way the most preferred object will be located on top. The indices that represent each donor (and graft) will be saved in the preference matrix in the same position as its corresponding score in the ordered points matrix; indices with ties in scores will be reordered randomly among their positions. The incompatibilities in each column will have same score but it is not necessary to randomly reorder them since the patient would stay with his initial donor rather than receiving an incompatible kidney. The code is detailed in *PreferenceMat.R*:

PreferenceMat.R

```
prefMat<-matrix(data=0,nCouples,nCouples,byrow=TRUE)

for(i in 1:nCouples){
  auxMat<-matrix(data=0,nCouples,2,byrow=TRUE)
  for(j in 1:nCouples){
    auxMat[j,1]=j
    auxMat[j,2]=pointsMat[j,i]
  }
  auxDataFr<-as.data.frame(auxMat)
  auxDataFr<-auxDataFr[order(auxDataFr$"V2", decreasing = TRUE),]

  #if there are still ties in some column we order them randomly.
    This will not be applied for objects with -2 points, since they
    will not take part in the problem because of incompatibilities,
    and therefore, it is not necessary.
  duplicates<-auxDataFr[duplicated(auxDataFr$"V2"),]
  auxVect<-c()
  for(p in duplicates[,"V2"]){
    if(is.element(p,auxVect)==FALSE & p!=-2){
      auxVect<-append(auxVect,p)
    }
  }
  if(length(auxVect)!=0){
    for(q in auxVect){
      auxInd<-c()
      for(l in 1:nCouples){
        if(auxDataFr[l,"V2"]==q){
          auxInd<-append(auxInd,auxDataFr[l,"V1"])
        }}
      auxIndRand=sample(auxInd)
      pos<-0
      for(m in 1:length(auxIndRand)){
        for(n in pos+1:nCouples){
```

```
          pos=n
          if(auxDataFr[n,"V2"]==q){
            auxDataFr[n,"V1"]=auxIndRand[m]
            break
          }}}}}

  for(k in 1:nCouples){
    prefMat[k,i]=auxDataFr[k,"V1"]
  }
}
```

Therefore, the preference matrix derived from the previous two examples is the one given in Example 10.

Example 10.

$$prefMat = \begin{pmatrix} P_{a_1} & P_{a_2} & P_{a_3} & P_{a_4} & P_{a_5} & P_{a_6} & P_{a_7} & P_{a_8} & P_{a_9} & P_{a_{10}} \\ 3 & 3 & 4 & 3 & 8 & 5 & 5 & 5 & 3 & 3 \\ 4 & 1 & 2 & 1 & 3 & 8 & 8 & 3 & 6 & 6 \\ 2 & 10 & 1 & 10 & 9 & 3 & 3 & 9 & 4 & 4 \\ 10 & 4 & 10 & 2 & 1 & 9 & 9 & 4 & 2 & 2 \\ 1 & 2 & 3 & 4 & 4 & 4 & 4 & 1 & 1 & 1 \\ 5 & 5 & 5 & 5 & 2 & 10 & 10 & 10 & 10 & 10 \\ 6 & 6 & 6 & 6 & 10 & 2 & 2 & 2 & 9 & 5 \\ 7 & 7 & 7 & 7 & 5 & 1 & 1 & 8 & 5 & 7 \\ 8 & 8 & 8 & 8 & 6 & 6 & 7 & 6 & 7 & 8 \\ 9 & 9 & 9 & 9 & 7 & 7 & 6 & 7 & 8 & 9 \end{pmatrix}$$

Subsections B.2.1 and B.2.2, and section B.3 propose randomly generated data and a preference matrix which preserve the particularities of the PID registry between 2009 and 2014.

Appendix C

Gale's TTC Algorithm Implementation Code

Given a preference matrix, the code for the implementation of TTC algorithm from Gale is shown in *TTC_ImplementationCode.cpp*:

TTC_ImplementationCode.cpp

```
// Gale's TTC algorithm implementation.
// Miren Lur Barquin Torre. 17 February 2020.

#include<iostream>
#include<vector>
#include<numeric>
#include<cmath>
#include<string>
#include<ctime>
#include<random>
#include<time.h>
#include<iomanip>
#include<fstream>

using namespace std;

void PreferenceMat();   // Initial preference matrix using vectors & number of couples.
vector<vector<int>> inCycle(int vertex, vector<int> AdjacVect);   // Returns a vector with 1 (true) if vertex is in a cycle (or loop) and 0 (false) if vertex is not in a cycle (or loop) as first element, and the cycle to which vertex belongs in second position (empty cycle if vertex does not belong to any).

clock_t timeUsed;
double cpu_time_used;
float nCouples;
int nStages = 0, nLoops = 0, nCycles = 0, lengthMaxCycles = 0,
```

```
    lengthMinCycles,
   nMinCycles, nMaxCycles = 0;
vector<int> V, auxNumCyclesInStage, auxLengthCycles;  // Vector V will represent at the same time agents, objects and vertices.
vector<vector<int>> PrefMat;

struct Results {
   int dim, num_Stages, num_Transplants, num_Cycles, numMin_CyclesInStage, numMax_CyclesInStage, lengthMin_Cycles, lenghtMax_Cycles;
   float time, numAverage_CyclesInStage, lengthAverage_Cycles, percent_Transpl;
};

int main() {
   PreferenceMat();  // Obtains the preference matrix & number of pairs taking part in the exchange.
   nMinCycles = nCouples;   // nCouples would be the max number of cycles that could be in the problem (considering all were loops).
   lengthMinCycles = nCouples;    // The longest possible cycle is the one in which every couple participate.
   timeUsed = clock();
   vector<int> FinalAllocAdjacVector(nCouples); // Creates a zero vector in which the final allocation will be saved.
   cout << "\nWHAT IS HAPPENING DURING THE RESOLUTION: ";
   int noncerosV = V.size();
   while (noncerosV != 0) {
      nStages++;
      cout << "\nSTAGE " << nStages;
      int nCyclesInThisStage = 0;
      vector<int> AdjacVect(nCouples);   //Creates a zero vector in which the adjacency vector at the current stage will be saved. Creating the adjacency vector for stage nStage, will be as drawing the graph.
      for (int j : V) {
         if (j != 0) {
            for (int i = 0; i < nCouples; i++) {
               if (PrefMat[i][j - 1] != 0) {
                  AdjacVect[j - 1] = PrefMat[i][j - 1];
                  break;
               }}}}

      //Once we have the graph with the arcs, we can search for the cycles and the loops.
      for (int k : V) {
         if (k != 0) {     // If k is still in the problem.
            if (inCycle(k, AdjacVect)[0][0] == true) {
               cout << "\n cycle(s): ";
```

```
                vector<int> cycle = inCycle(k, AdjacVect)[1];
                for (int i : cycle) {
                   cout << i << " ";
                }
                // See if it is a loop or a cycle.
                int lengthCycle = cycle.size();
                if (lengthCycle == 1) {
                   nLoops++;
                }
                else {
                   auxLengthCycles.push_back(lengthCycle);
                   if (lengthCycle > lengthMaxCycles) { lengthMaxCycles = lengthCycle; }
                   if (lengthCycle < lengthMinCycles) { lengthMinCycles = lengthCycle; }
                   nCycles++;
                   nCyclesInThisStage++;
                }
                // Allocate to each agent taking part in the cycle the corresponding object.
                for (int v : cycle) {
                   FinalAllocAdjacVector[v - 1] = AdjacVect[v - 1];
                   for (int i = 0; i < nCouples; i++) {
                      //Delete the column corresponding to agent v from the Preference matrix.
                      PrefMat[i][v - 1] = 0;
                   }
                   for (int j : V) {
                      if (j != 0) {
                         if (j != v) {
                            // Delete the object v from the preferences of the agents still in the problem
                            for (int i = 0; i < nCouples; i++) {
                               if (PrefMat[i][j - 1] == v) {
                                  PrefMat[i][j - 1] = 0;
                                  break;
                               }}}}}

                   // Remove the vertices in the cycle from V because they have already been assigned.
                   for (int i = 0; i < V.size(); i++) {
                      if (V[i] == v) {
                         V[i] = 0;
                         noncerosV--;
                         break;
                      }}}}}}

      auxNumCyclesInStage.push_back(nCyclesInThisStage);
      if (nCyclesInThisStage > nMaxCycles) { nMaxCycles =
```

```
nCyclesInThisStage; }
        if (nCyclesInThisStage < nMinCycles) { nMinCycles = nCyclesInThisStage; }
    }
    cout << "\n \n \n The final allocation vector will be:\n";
    cout << "( ";
    for (int i : FinalAllocAdjacVector) {
        cout << i << " ";
    }
    cout << ")";
    cout << "\n \n";
    cout << "FINAL ALLOCATION: " << endl;
    for (int i = 0; i < nCouples; i++) {
        cout << "Agent: " << i + 1 << " obtains object: " << FinalAllocAdjacVector[i] << "." << endl;
    }
    cout << "* Keep in mind that in those cases where the agent obtains its initial object, the transplant will not be carried out. ";

    if(auxLengthCycles.size() == 0) {auxLengthCycles.push_back(0); lengthMinCycles = 0; } // In the case that there are no cycles (just loops)
    if(auxNumCyclesInStage.size() == 0) {auxNumCyclesInStage.push_back(0); nMinCycles = 0; }

    float numAverageCyclesInStage = accumulate(auxNumCyclesInStage.begin(), auxNumCyclesInStage.end(), 0.0) / auxNumCyclesInStage.size();
    float AverageLength = accumulate(auxLengthCycles.begin(), auxLengthCycles.end(), 0.0) / auxLengthCycles.size();

    timeUsed = clock() - timeUsed;
    cpu_time_used = ((double)(timeUsed)) / CLOCKS_PER_SEC;

    Results FinalResults;
    FinalResults.dim = nCouples;
    FinalResults.num_Stages = nStages;
    FinalResults.time = cpu_time_used;
    FinalResults.num_Transplants = nCouples - nLoops;
    FinalResults.percent_Transpl = ((nCouples - nLoops) * 100) / nCouples;
    FinalResults.num_Cycles = nCycles;
    FinalResults.numMax_CyclesInStage = nMaxCycles;
    FinalResults.numAverage_CyclesInStage = numAverageCyclesInStage;
    FinalResults.numMin_CyclesInStage = nMinCycles;
    FinalResults.lenghtMax_Cycles = lengthMaxCycles;
    FinalResults.lengthAverage_Cycles = AverageLength;
    FinalResults.lengthMin_Cycles = lengthMinCycles;
```

```

    cout << "\n \n";
    cout << "RESULTS: " << endl;
    cout << "Dimension: " << FinalResults.dim << endl;
    cout << "Number of stages: " << FinalResults.num_Stages << endl;
    cout << "time: " << FinalResults.time << endl;
    cout << "Number of transplants: " << FinalResults.num_Transplants << endl;
    cout << "Percentage of transplanted kidneys: " << FinalResults.percent_Transpl << endl;
    cout << "Number of cycles: " << FinalResults.num_Cycles << endl;
    cout << "Minimum number of cycles in a stage: " << FinalResults.numMin_CyclesInStage << endl;
    cout << "Average number of cycles in a stage: " << FinalResults.numAverage_CyclesInStage << endl;
    cout << "Maximum number of cycles in a stage: " << FinalResults.numMax_CyclesInStage << endl;
    cout << "Length of the shortest cycle: " << FinalResults.lengthMin_Cycles << endl;
    cout << "Average length of cycles: " << FinalResults.lengthAverage_Cycles << endl;
    cout << "Length of the longest cycle : " << FinalResults.lenghtMax_Cycles << endl;

    ofstream out("ResultadosTTCG.txt", fstream::app);
    out <<FinalResults.dim << " & " << FinalResults.num_Stages << " & " << FinalResults.time << " & " << FinalResults.num_Transplants << " & " << FinalResults.percent_Transpl << " & " << FinalResults.num_Cycles << " & " << FinalResults.numMin_CyclesInStage << " & " << FinalResults.numAverage_CyclesInStage << " & " << FinalResults.numMax_CyclesInStage << " & " << FinalResults.lengthMin_Cycles << " & " << FinalResults.lengthAverage_Cycles << " & " << FinalResults.lenghtMax_Cycles <<" \\\\ "<< endl;
    out << "\\hline" << endl;
    out.close();
    return 0;
}

// DEFINITION OF FUNCTIONS:

void PreferenceMat() {
    // See how many couples are taking part in the problem.
    int pairs = 0;
    ifstream nPairs("RPrefMat.dat", fstream::binary | fstream::out);
    string unused;
    while (getline(nPairs, unused))
        ++pairs;
    nPairs.close();
```

```

    // Obtain the preference Matrix.
    nCouples = pairs;
    for (int i = 0; i < nCouples; i++) {
        V.push_back(i + 1);}

    vector<vector<int>> AuxPrefMat(nCouples, vector<int>(nCouples));
    ifstream mat("RPrefMat.dat", fstream::binary | fstream::out);
    for (int i = 0; i < nCouples; i++) {
        for (int j = 0; j < nCouples; j++) {
            mat >> AuxPrefMat[i][j];}}

    mat.close();
    PrefMat = AuxPrefMat;

    //SHOW THE MATRIX:
    cout << "\n \n The preference matrix is: \n";
    cout << fixed << setfill(' ');
    for (int i = 0; i < nCouples; i++) {
        for (int j = 0; j < nCouples; j++) {
            cout << setw(5) << PrefMat[i][j];}
        cout << "\n";}
}

vector<vector<int>> inCycle(int vertex, vector<int> AdjacVect) {
    int aux = AdjacVect[vertex - 1];
    bool IsInAuxVect = false;
    vector<int> auxVect, cycle;
    if (vertex != aux) { auxVect.push_back(vertex); }
    auxVect.push_back(aux);
    while (IsInAuxVect == false) {
        if (AdjacVect[aux - 1] == vertex) {
            cycle = auxVect;
            vector<vector<int>> ToReturn = { {1}, cycle };
            return ToReturn;}
        else {
            aux = AdjacVect[aux - 1];
            for (int i : auxVect) {
                if (i == aux) {
                    IsInAuxVect = true;
                    break;
                }}
            if (IsInAuxVect != true) {
                auxVect.push_back(aux);
            }}}

    vector<vector<int>> ToReturn = { {0}, {} };
    return ToReturn;
}
```

Appendix D

Gale's TTC Algorithm Implementation Results

To study the performance of the TTC algorithm from Gale, let us take a sample size of 20 simulations for some fixed numbers of patient-donor couples taking part in the problem; that is, for 5, 10, 20, 50, 100, 200, and 350 couples we will run 20 simulations. The results are collected in Table 16, where the information is organised in columns as follows:

First column: (n) number of couples taking part in the problem.

Second column: (St.) number of stages.

Third column: (Time) seconds to solve each problem.

Fourth column:(Tr.) number of possible transplants.

Fifth column: (%Tr.) percentage of possible transplants.

Sixth column: (Cy.) number of cycles.

Seventh, eighth, and ninth columns: (min./avg./max. Cy/St) minimum, average, and maximum number of cycles in a stage, respectively.

Tenth, eleventh, and twelfth columns: (min./avg./max. Len. Cy.) minimum, average, and maximum length of a cycle in a stage, respectively.

Notice that the running time on the third column measures the seconds it takes the program to apply Gale's TTC algorithm to each previously generated preference matrix, without considering the time it takes to generate the data, points and preference matrices.

Table 16: Results of simulations for $n = 5, 10, 20, 50, 100, 200, 350$.

n	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
5	3	0.01	2	40	1	0	0.33	1	2	2	2
5	4	0.01	2	40	1	0	0.25	1	2	2	2
5	4	0.01	2	40	1	0	0.25	1	2	2	2
5	2	0.01	2	40	1	0	0.50	1	2	2	2
5	2	0.01	5	100	2	1	1	1	2	2.50	3
5	3	0.01	3	60	1	0	0.33	1	3	3	3
5	4	0.01	0	0	0	0	0	0	0	0	0
5	4	0.01	0	0	0	0	0	0	0	0	0
5	2	0.01	0	0	0	0	0	0	0	0	0
5	2	0.01	0	0	0	0	0	0	0	0	0
5	3	0.00	4	80	2	0	0.67	1	2	2	2
5	4	0.01	2	40	1	0	0.25	1	2	2	2
5	3	0.01	2	40	1	0	0.33	1	2	2	2
5	2	0.00	2	40	1	0	0.50	1	2	2	2
5	3	0.00	2	40	1	0	0.33	1	2	2	2
5	3	0.01	0	0	0	0	0	0	0	0	0
5	3	0.01	4	80	2	0	0.67	1	2	2	2
5	3	0.00	2	40	1	0	0.33	1	2	2	2
5	3	0.01	2	40	1	0	0.33	1	2	2	2
5	3	0.01	2	40	1	0	0.33	1	2	2	2
10	6	0.05	6	60	3	0	0.50	1	2	2	2
10	4	0.03	5	50	2	0	0.50	1	2	2.50	3
10	5	0.03	4	40	2	0	0.40	1	2	2	2
10	4	0.02	4	40	2	0	0.50	1	2	2	2
10	4	0.02	4	40	2	0	0.50	1	2	2	2
10	5	0.03	2	20	1	0	0.20	1	2	2	2
10	5	0.02	9	90	4	0	0.80	1	2	2.25	3
10	6	0.05	8	80	4	0	0.67	1	2	2	2
10	5	0.04	0	0	0	0	0	0	0	0	0
10	5	0.03	7	70	3	0	0.60	1	2	2.33	3
10	4	0.05	8	80	3	0	0.75	1	2	2.67	4
10	5	0.03	2	20	1	0	0.20	1	2	2	2
10	4	0.02	9	90	3	0	0.75	1	2	3	4
10	3	0.03	8	80	3	1	1	1	2	2.67	4
10	5	0.03	4	40	2	0	0.40	1	2	2	2
10	3	0.02	8	80	4	1	1.33	2	2	2	2
10	4	0.03	6	60	3	0	0.75	2	2	2	2
10	4	0.05	4	40	2	0	0.50	1	2	2	2
10	6	0.04	6	60	3	0	0.50	1	2	2	2
10	4	0.03	7	70	3	0	0.75	1	2	2.33	3
20	8	0.04	16	80	7	0	0.88	1	2	2.29	3
20	7	0.04	15	75	7	0	1	2	2	2.14	3
20	8	0.05	11	55	5	0	0.63	1	2	2.20	3
20	9	0.06	14	70	6	0	0.67	1	2	2.33	3
20	9	0.05	17	85	8	0	0.89	1	2	2.13	3

Continued on next page

n	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
20	7	0.05	7	35	3	0	0.43	1	2	2.33	3
20	6	0.06	9	45	3	0	0.50	1	3	3	3
20	7	0.06	11	55	4	0	0.57	1	2	2.75	4
20	9	0.06	16	80	7	0	0.78	1	2	2.29	3
20	8	0.06	12	60	5	0	0.63	1	2	2.40	4
20	9	0.05	11	55	5	0	0.56	1	2	2.20	3
20	9	0.04	13	65	5	0	0.56	1	2	2.60	5
20	8	0.07	11	55	5	0	0.63	1	2	2.20	3
20	9	0.05	16	80	6	0	0.67	1	2	2.67	4
20	9	0.04	15	75	7	0	0.78	1	2	2.14	3
20	8	0.06	13	65	6	0	0.75	1	2	2.17	3
20	9	0.06	12	60	5	0	0.56	1	2	2.40	4
20	8	0.06	18	90	7	0	0.88	1	2	2.57	4
20	6	0.06	7	35	3	0	0.50	1	2	2.33	3
20	8	0.05	16	80	6	0	0.75	1	2	2.67	4
50	15	0.11	38	76	14	0	0.93	2	2	2.71	5
50	17	0.15	37	74	14	0	0.82	2	2	2.64	5
50	17	0.17	38	76	16	0	0.94	2	2	2.38	4
50	17	0.17	33	66	16	0	0.94	2	2	2.06	3
50	13	0.13	35	70	14	0	1.08	2	2	2.50	4
50	18	0.10	38	76	16	0	0.89	2	2	2.38	4
50	18	0.12	46	92	19	0	1.06	2	2	2.42	4
50	18	0.14	33	66	15	0	0.83	2	2	2.20	3
50	18	0.15	36	72	16	0	0.89	1	2	2.25	3
50	15	0.17	28	56	13	0	0.87	2	2	2.15	3
50	18	0.13	41	82	17	0	0.94	2	2	2.41	4
50	19	0.16	33	66	15	0	0.79	2	2	2.20	4
50	14	0.10	34	68	13	0	0.93	2	2	2.62	4
50	15	0.14	30	60	13	0	0.87	2	2	2.31	4
50	14	0.13	28	56	12	0	0.86	2	2	2.33	4
50	14	0.12	33	66	13	0	0.93	2	2	2.54	4
50	15	0.11	38	76	15	0	1	3	2	2.53	5
50	16	0.11	42	84	14	0	0.88	1	2	3	5
50	17	0.14	34	68	14	0	0.82	1	2	2.43	4
50	17	0.12	43	86	17	0	1	2	2	2.53	4
100	31	0.22	83	83	31	0	1	2	2	2.68	5
100	25	0.29	68	68	26	0	1.04	2	2	2.62	5
100	26	0.32	76	76	28	0	1.08	3	2	2.71	5
100	30	0.24	75	75	32	0	1.07	2	2	2.34	4
100	31	0.21	79	79	32	0	1.03	2	2	2.47	5
100	35	0.31	78	78	34	0	0.97	3	2	2.29	5
100	31	0.27	85	85	29	0	0.94	2	2	2.93	5
100	31	0.24	80	80	31	0	1	2	2	2.58	5
100	36	0.38	92	92	38	0	1.06	2	2	2.42	4
100	30	0.22	83	83	30	0	1	2	2	2.77	5
100	32	0.28	79	79	32	0	1	2	2	2.47	5
100	28	0.34	70	70	25	0	0.89	2	2	2.80	6

Continued on next page

n	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
100	28	0.19	62	62	24	0	0.86	2	2	2.58	5
100	27	0.29	78	78	29	0	1.07	2	2	2.69	6
100	27	0.29	65	65	24	0	0.89	1	2	2.71	6
100	30	0.26	78	78	32	0	1.07	3	2	2.44	6
100	23	0.26	74	74	27	0	1.17	2	2	2.74	5
100	32	0.41	69	69	28	0	0.88	2	2	2.46	4
100	31	0.38	72	72	28	0	0.90	3	2	2.57	6
100	25	0.21	64	64	27	0	1.08	2	2	2.37	5
200	56	0.79	159	79.5	58	0	1.04	2	2	2.74	8
200	60	0.77	148	74	65	0	1.08	2	2	2.28	5
200	54	1.07	148	74	55	0	1.02	3	2	2.69	9
200	52	0.99	152	76	57	0	1.10	3	2	2.67	7
200	50	0.76	144	72	56	0	1.12	4	2	2.57	8
200	58	1.13	157	78.5	64	0	1.10	3	2	2.45	4
200	50	0.59	145	72.5	56	0	1.12	3	2	2.59	4
200	48	0.58	141	70.5	52	0	1.08	2	2	2.71	5
200	54	0.69	161	80.5	60	0	1.11	2	2	2.68	6
200	52	1.03	149	74.5	56	0	1.08	2	2	2.66	6
200	45	0.59	147	73.5	58	0	1.29	4	2	2.53	6
200	47	0.88	133	66.5	49	0	1.04	2	2	2.71	5
200	44	0.90	135	67.5	50	0	1.14	2	2	2.70	6
200	56	0.60	142	71	57	0	1.02	2	2	2.49	5
200	54	0.54	153	76.5	58	0	1.07	3	2	2.64	6
200	55	1.02	154	77	60	0	1.09	3	2	2.57	6
200	53	0.51	169	84.5	62	0	1.17	3	2	2.73	6
200	56	0.55	146	73	56	0	1	2	2	2.61	6
200	50	0.57	141	70.5	61	0	1.22	3	2	2.31	5
200	55	0.65	184	92	65	0	1.18	3	2	2.83	6
350	84	1.64	261	74.57	93	0	1.11	4	2	2.81	9
350	88	1.72	269	76.86	98	0	1.11	3	2	2.75	6
350	93	2.54	275	78.57	105	0	1.13	3	2	2.62	5
350	91	1.80	293	83.71	107	0	1.18	4	2	2.74	7
350	84	1.70	276	78.86	95	0	1.13	4	2	2.91	8
350	87	1.70	288	82.29	108	0	1.24	3	2	2.67	7
350	85	1.81	259	74	103	0	1.21	4	2	2.52	7
350	81	2.29	253	72.29	98	0	1.21	4	2	2.58	6
350	87	1.47	261	74.57	100	0	1.15	5	2	2.61	7
350	85	1.66	293	83.71	106	0	1.25	4	2	2.76	8
350	90	1.71	268	76.57	95	0	1.06	3	2	2.82	7
350	85	1.52	277	79.14	104	0	1.22	3	2	2.66	6
350	74	1.50	235	67.14	87	0	1.18	5	2	2.70	6
350	87	1.73	261	74.57	100	0	1.15	3	2	2.61	6
350	81	2.29	267	76.29	96	0	1.19	3	2	2.78	8
350	93	1.60	283	80.86	111	0	1.20	3	2	2.55	6
350	87	1.59	274	78.29	105	0	1.21	3	2	2.61	6
350	72	1.39	235	67.14	83	0	1.15	3	2	2.83	9
350	82	1.83	276	78.86	98	0	1.20	3	2	2.82	9
350	86	2.22	274	78.29	98	0	1.14	3	2	2.80	9

R-squared measures of goodness of fit for some regression models, where the dependent variable is the number of patient-donor couples n, are shown in Table 17. The highest values are in bold.

Table 17: R-squared measures

Regression	St.	Time	Tr.	%Tr.	Cy.	Cy/St			Len. Cy.		
						min.	avg.	max.	min.	avg.	max.
Linear	0.997	0.973	**1.000**	0.414	**1.000**	0.230	0.653	0.912	0.143	0.464	0.846
Power	**0.998**	**0.978**	0.995	0.763	0.995	–	0.885	**0.975**	0.448	0.777	0.969
Inverse	0.405	0.272	0.364	0.994	0.378	0.466	0.886	0.591	**0.861**	0.981	0.728
Exponential	0.804	0.767	0.733	0.357	0.734	–	0.508	0.770	0.143	0.405	0.674
Logarithmic	0.818	0.654	0.776	0.827	0.791	**0.487**	**0.971**	0.938	0.450	0.831	**0.987**
S-curve	0.744	0.826	0.833	**0.998**	0.831	–	0.969	0.747	0.860	**0.986**	0.892

–: not calculable.

The best fitted regression curves for each variable are as follows:

$$St. = 0.762 \cdot n^{0.797}$$
$$Time = 0.001 \cdot n^{1.251}$$
$$Tr. = -2.397 + 0.773 \cdot n$$
$$\%Tr. = e^{4.351 - 3.523 \cdot n^{-1}}$$
$$Cy. = 0.034 + 0.286 \cdot n$$
$$min.Cy/St = 0.088 - 0.018 \cdot ln(n)$$
$$avg.Cy/St = 0.090 + 0.193 \cdot ln(n)$$
$$max.Cy/St = 0.426 \cdot n^{0.354}$$
$$min.Len.Cy = 2.051 - 2.213 \cdot n^{-1}$$
$$avg.Len.Cy = e^{0.979 - 2.580 \cdot n^{-1}}$$
$$max.Len.Cy = -0.472 + 1.236 \cdot ln(n)$$

References

[1] Abdulkadiroğlu, A., & Sönmez, T. (1999). House allocation with existing tenants. Journal of Economic Theory, 88(2), 233-260.

[2] Abdulkadiroğlu, A., & Sönmez, T. (2003). School choice: A mechanism design approach. American economic review, 93(3), 729-747.

[3] Appa, G. M. (1973). The transportation problem and its variants. Journal of the Operational Research Society, 24(1), 79-99.

[4] Burkard, R., Dell'Amico, M., & Martello, S. (2012). Assignment Problems. Revised reprint. Other Titles in Applied Mathematics, 106.

[5] Domínguez-Gil, B., de la Oliva Valentín, M., Martín Escobar, E., Cruzado, J. M., Pascual, J., & Fernández Fresnedo, G. (2010). *Situación actual del trasplante renal de donante vivo en España y otros países: pasado, presente y futuro de una excelente opción terapéutica*. Nefrología, 30, 3-13.

[6] EFE, R. T. V. E. (2019). España, veintisiete años líder mundial en donación y trasplante de órganos. Retrieved January 15, 2020, from `http://www.rtve.es/noticias/20190114/espana-veintisiete-anos-lider-mundial-donacion-trasplante-organos/1868920.shtml`.

[7] Gebel, H. M., Bray, R. A., & Nickerson, P. (2003). Pre-transplant assessment of donor-reactive, HLA-specific antibodies in renal transplantation: contraindication vs. risk. American Journal of Transplantation, 3(12), 1488-1500.

[8] Hylland, A., & Zeckhauser, R. (1979). The efficient allocation of individuals to positions. Journal of Political economy, 87(2), 293-314.

[9] Ibaraki, T., & Katoh, N. (1988). Resource allocation problems: algorithmic approaches. MIT press. 992-1023

[10] Interterritorial Council of the National Health Service of Spain. Retrieved January 20, 2020, from `http://www.ont.es/infesp/Paginas/ConsejoInterritorial.aspx`.

[11] ISO/IEC. (2017). ISO International Standard ISO/IEC 14882:2017(E) – Programming Language C++. Geneva, Switzerland: International Organization for Standardization (ISO). Retrieved November 20, 2019, from `https://isocpp.org/std/the-standard`.

[12] Jefatura del Estado. (1979). Agencia Estatal Boletín Oficial del Estado. Ley 30/1979, de 27 de octubre, sobre extracción y trasplante de órganos. BOE, 266, 25742-25743. Ref.: BOE-A-1979-26445.

[13] Jefatura del Estado. (1999). Agencia Estatal Boletín Oficial del Estado. Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal. BOE, 298(2), 43088-43099. Ref.: BOE-A-1999-23750.

[14] Jungnickel, D. (2013). *Basic graph theory.* Graphs, Networks and Algorithms. Algorithms and Computation in Mathematics, 5, 1-33.

[15] Keizer, K. M., de Klerk, M., Haase-Kromwijk, B. J. J. M., & Weimar, W. (2005). The Dutch algorithm for allocation in living donor kidney exchange. In Transplantation proceedings, 37(2), 589-591.

[16] Ma, J. (1994). Strategy-proofness and the strict core in a market with indivisibilities. International Journal of Game Theory, 23(1), 75-83.

[17] Mandal, A. K., Snyder, J. J., Gilbertson, D. T., Collins, A. J., & Silkensen, J. R. (2003). Does cadaveric donor renal transplantation ever provide better outcomes than live-donor renal transplantation? 1, 2. Transplantation, 75(4), 494-500.

[18] Massó, J. (2010). *El intercambio de riñones y la Matemática Discreta.* Paseo por la Geometría 2009-2010, Facultad de Ciencias de la Universidad del País Vasco.

[19] Massó, J. (2015). The theory of stable allocations and the practice of market design. The Nobel Prize in Economics 2012 for Alvin E. Roth and Lloyd S. Shapley. Contributions to Science, 11, 103-112.

[20] Massó, J. (2019). Professor of Economics at the Universitat Autónoma de Barcelona & PhD.: State University of New York at Stony Brook (1988). Retrieved January 28, 2020, from `http://pareto.uab.es/jmasso/index.html`

[21] Muñoz, M. O. V. (2016). Puesta en marcha de un Plan Nacional de Donación Cruzada y cadenas de trasplante en España: expectativas

de la donación altruista (Doctoral dissertation, Complutense University of Madrid). Retrieved January 19, 2020, from https://eprints.ucm.es/44633/1/T39276.pdf.

[22] Organización Nacional de Trasplantes. (2020). Retrieved January 20, 2020, from http://www.ont.es.

[23] Organización Nacional de Trasplantes. (2011). Ministerio de Sanidad, Consumo y Bienestar Social. *España realiza con éxito la primera cadena de trasplantes con donante altruista* [Press release]. Retrieved January 12, 2020, from http://www.ont.es/prensa/NotasDePrensa/280411%20Espa%C3%B1a%20realiza%20con%20%C3%A9xito%20la%20primera%20cadena%20de%20trasplantes%20con%20donante%20altruista.pdf.

[24] Organización Nacional de Trasplantes. (2014). Ministerio de Sanidad, Consumo y Bienestar Social. *España realiza por primera vez una cadena de 6 trasplantes renales de vivo, con donante "samaritano", en varios tiempos* [Press release]. Retrieved December 12, 2019, from http://www.ont.es/prensa/NotasDePrensa/08%2005%202014%20%20Trasplantes%20cadena.pdf.

[25] Organización Nacional de Trasplantes. (2015). Ministerio de Sanidad, Consumo y Bienestar Social. *Programa Nacional de Donación Cruzada en España* [Consensus document]. Retrieved January 19, 2020, from http://www.ont.es/infesp/DocumentosDeConsenso/Programa%20Donaci%C3%B3n%20Renal%20Cruzada_actualizaci%C3%B3n_25062015.pdf.

[26] Organización Nacional de Trasplantes. (2018). Ministerio de Sanidad, Consumo y Bienestar Social. *Memoria actividad donación y trasplante renal. España 2018.* Retrieved October 4, 2019, from http://www.ont.es/infesp/Memorias/Actividad%20de%20Donaci%C3%B3n%20y%20Trasplante%20Renal.pdf.

[27] Organización Nacional de Trasplantes. (2018). Ministerio de Sanidad, Consumo y Bienestar Social. *La ONT, las Sociedades Científicas y los pacientes trasplantados apuestan por la donación renal de vivo* [Press release]. Retrieved December 13, 2019, from http://www.ont.es/Documents/06.06.2018%20NP%20D%C3%ADa%20Nacional%20del%20Donante_.pdf.

[28] Organización Nacional de Trasplantes. (2018). Ministerio de Sanidad, Consumo y Bienestar Social. *España e Italia protagonizan el primer trasplante renal cruzado internacional del sur de Europa* [Press release]. Retrieved December 13, 2019, from http://www.ont.es

/prensa/NotasDePrensa/2018%2008%2009%20-%20Espa%C3%B1a%20e%20Italia%20protagonizan%20el%20primer%20trasplante%20renal%20cruzado%20internacional.%20(Espa%C3%B1ol).pdf.

[29] Organización Nacional de Trasplantes. (2019). Ministerio de Sanidad, Consumo y Bienestar Social. *El Registro Mundial de Trasplantes cifra en 139.024 los trasplantes realizados en el mundo en el último año, con un aumento del 2,3%* [Press release]. Retrieved December 12, 2019, from http://www.ont.es/prensa/NotasDePrensa/28%2008%202019%20%20REGISTRO%20MUNDIAL%20DE%20TRASPLANTES.pdf.

[30] Organización Nacional de Trasplantes. (2019). Ministerio de Sanidad, Consumo y Bienestar Social. *España e Italia protagonizan el segundo trasplante renal cruzado internacional del sur de Europa* [Press release]. Retrieved December 13, 2019, from http://www.ont.es/prensa/NotasDePrensa/06.11.2019%20NP%20Espa%C3%B1a%20e%20Italia%20protagonizan%20el%20segundo%20trasplante%20cruzado%20internacional%20OK(1).pdf.

[31] R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. Retrieved January 12, 2020, from http://www.R-project.org/.

[32] Rees, M.A., Kopke, J.E., Pelletier, R.P., Segev, D.L., Rutter,M.E., Fabrega, A.J., Rogers, J., Pankewycz, O.G., Hiller, J., Roth, A.E., Sandholm, T., Ünver, U., and Montgomery, R.A. (2009). A non-simultaneous, extended, altruistic-donor chain. The New England Journal of Medicine, 360, 1096-1101.

[33] Revuelta del Campo, F. (2018). Estudio y análisis de algoritmos de optimización para la obtención de cadenas de cruce óptimas en trasplantes cruzados (Final Degree Dissertation, Polytechnic University of Madrid). Retrieved January 19, 2020, from http://oa.upm.es/50343/1/TFG_FERNANDO_REVUELTA_DEL_CAMPO.pdf.

[34] Roth, A. E. (1982). Incentive compatibility in a market with indivisible goods. Economics letters, 9(2), 127-132.

[35] Roth, A. E. (2018). The Craig and Susan McCaw Professor of Economics at Stanford University. Retrieved January 29, 2020, from http://web.stanford.edu/~alroth/.

[36] Roth, A. E., & Postlewaite, A. (1977). Weak versus strong domination in a market with indivisible goods. Journal of Mathematical Economics, 4(2), 131-137.

[37] Roth, A. E., Sönmez, T., & Ünver, M. U. (2004). Kidney exchange. The Quarterly Journal of Economics, 119(2), 457-488.

[38] Shankarkumar, U. (2004). The human leukocyte antigen (HLA) system. International Journal of Human Genetics, 4(2), 91-103. Retrieved January 20, 2020, from `http://www.krepublishers.com/02-Journals/IJHG/IJHG-04-0-000-000-2004-Web/IJHG-04-2-091-166-2004-Abst-PDF/IJHG-04-2-090-103-2004-Shankar/IJHG-04-2-091-103-2004-Shankar.pdf`.

[39] Shapley, L., & Scarf, H. (1974). *On cores and indivisibility.* Journal of mathematical economics, 1(1), 23-37.

[40] Sönmez, T., & Ünver, M. U. (2010). House allocation with existing tenants: A characterization. Games and Economic Behavior, 69(2), 425-445.

[41] Starzl, T. E. (2001). Felix T. Rapaport, MD. JAMA, 285(23), 3032.

[42] The Royal Swedish Academy Of Sciences. (2012). The Prize in Economic Sciences 2012. Retrieved January 22, 2020, from `https://www.nobelprize.org/prizes/economic-sciences/2012/summary/`.

[43] Ünver, M. U. (2010). Dynamic kidney exchange. The Review of Economic Studies, 77(1), 372-414.

[44] Ünver, M. U. (2020). Professor of Economics at Boston College. Retrieved January 29, 2020, from `https://sites.bc.edu/utku-unver/`.

[45] Wako, J. (1984). A note on the strong core of a market with indivisible goods. Journal of Mathematical Economics, 13(2), 189-194.

[46] Wako, J. (1991). Some properties of weak domination in an exchange market with indivisible goods. The Economic studies quarterly, 42(4), 303-314.

[47] Weisstein, Eric W. "Graph Cycle." MathWorld–A Wolfram Web Resource. Retrieved February 11, 2020, from `http://mathworld.wolfram.com/GraphCycle.html`.

[48] Weisstein, Eric W. "Graph Edge." MathWorld–A Wolfram Web Resource. Retrieved February 11, 2020, from `http://mathworld.wolfram.com/GraphEdge.html`.

[49] Weisstein, E. W. "Maximum Independent Edge Set." MathWorld–A Wolfram Web Resource. Retrieved February 11, 2020, from `http://mathworld.wolfram.com/MaximumIndependentEdgeSet.html`.

[50] Weisstein, E. W. "Weighted Graph." MathWorld–A Wolfram Web Resource. Retrieved February 11, 2020, from http://mathworld.wolfram.com/WeightedGraph.html.

[51] World Health Organization. Retrieved January 20, 2020, from https://www.who.int.