# MATCHING MECHANISMS FOR KIDNEY TRANSPLANTATIONS

### SOLUTION

## 1. Introduction

Transplantation is the preferred treatment for most serious kidney diseases. Traditionally, kidneys are obtained from cadavers. The demand is however increasing more rapidly than the offer and more and more patients are waiting for a cadaver's kidney. In France in 2017, about 500 patients in the waiting list died because they have not been transplanted on time[1].

An alternative is to obtain kidneys from live donors. It is attractive because the probability of successful transplantation is higher. If a patient knows someone that is ready to donate him his kidney (a spouse, a parent, a friend) and if the transplant is feasible (donor and patient are immunologically compatible), the transplant is realized. If the donor is not compatible with his patient, the pair may enter a kidney exchange program. In most countries, there is a consensus to forbid organ selling, so that non-monetary incentive mechanisms should be designed for an efficient allocation of kidneys to patients. This is the subject of this project.

This subject as well as test files are available on this page:
`https://marceaucoupechoux.wp.imt.fr/enseignement/english-inf421-pi/`.
Test files encode the number of vertices $n$, the threshold $K$ and the adjacency matrix of the directed graph.

## 2. Model

A *kidney exchange problem* consists of a set of donor-patient pairs $N = \{(k_1, t_1), ..., (k_n, t_n)\}$, a set of compatible kidneys $K_i \subset K = \{k_1, ..., k_n\}$ for each patient $t_i$ and a strict preference relation $P_i$ over $K_i \cup \{k_i, w\}$ for each patient $t_i$[2]. For a patient $t_i$, $w$ denotes the option of entering the waiting list for cadavers' kidneys[3]. Option $k_i$ either means that $k_i$ is compatible with $t_i$, in which case $k_i$ is the most preferred option, or that $t_i$ prefers to not participate to the exchange rather than receiving a kidney with lower preference. We also assume that there is a strict priority list of the patients[4].

The *outcome* of the problem is a matching of patients to either kidneys or to the waiting list $w$. No kidney can be assigned to more than one patient; the waiting list can be matched to several patients.

A *mechanism* selects a matching for each kidney exchange problem.

[1] Agence de biomédecine.

[2] Preferences of a patient reflect the probability of successful transplant and may depend on several factors like closeness of tissue type, age, address of the donor, etc. How these preferences are built is out of the scope of the project.

[3] In exchange of a kidney $k_i$ to another pair, patient $t_i$ may benefit from a certain priority in the waiting list. This aspect is not considered in this project. The benefice is only reflected in the position of $w$ in the preference list of $t_i$.

[4] This list is established by organ banks for the allocation of cadaver organs.

**Definition 1** (Efficiency)**.** *A matching is* **Pareto efficient** *if there is no other matching that is weakly preferred by all patients and donors and strictly preferred by at least one patient-donor pair. A mechanism is* **efficient** *if it always selects a Pareto efficient matching.*

**Definition 2** (Strategy-proof)**.** *A mechanism is* **strategy-proof** *if a patient can never benefit by unilaterally misrepresenting his preferences.*

## 3. Two Simple Approaches

In this section, we consider two simple approaches for the matching of kidneys to patients. The first scheme can possibly be run as a preprocessing step before other matching mechanisms. The second one allows exchanges only between two pairs.

3.1. **Direct Donation Algorithm.** Kidney of a donor $k_i$ is assigned to its paired patient $t_i$ if they are compatible, i.e., if $k_i \in K_i$. Otherwise, $t_i$ is placed in the waiting list $w$.

**Question 1.** Implement the direct donation algorithm. Give the pseudo-code of your algorithm. Explain your implementation choices.

3.2. **Greedy Matching.** Let $G(V,E)$ be an undirected graph where the set of vertices $V$ is the set of donor-patient pairs $N$ and $\{(k_i,t_i),(k_j,t_j)\} \in E$ iff $k_j \in K_i$ and $k_i \in K_j$. We look for a maximum matching in $G$, i.e., a matching of maximum cardinality. There exists a polynomial-time algorithm (the blossom algorithm) that optimally solves this problem and that has been found by Edmond in 1961. To keep things simple here, we rely on a greedy algorithm.

**Question 2.** Implement the greedy algorithm. Your solution should take into account the priority list of patients and their preferences. Give the pseudo-code of your algorithm. Explain your implementation choices.

If we ignore the priority list and patient preferences, the greedy algorithm has an approximation ratio of 2, i.e., the size of an optimal matching (obtained e.g. with the blossom algorithm) is less than 2 times the size of a matching returned by the greedy algorithm. If priority lists and preferences are taken into account, the greedy algorithm is both Pareto efficient and strategy-proof.

We can improve the efficiency by considering longer cycles of exchanges and chains of patients.

## 4. Efficient Strategy-Proof Exchange Mechanism

In this section, we take into account patient's preferences and priority list.

4.1. **Preliminaries.** We consider a directed graph whose vertices are the kidneys, the patients and a waiting list $w$.

**Definition 3** (Cycle)**.** *A cycle is an ordered list of kidneys and patients* $(k'_1, t'_1, \ldots, k'_m, t'_m)$*, in which* $k'_1$ *points to* $t'_1$*,* $t'_1$ *points to* $k'_2$*,...,* $k'_m$ *points to* $t'_m$ *and* $t'_m$ *points to* $k'_1$*.*

**Definition 4** ($w$-Chain)**.** *A $w$-chain is an ordered list of kidneys and patients $(k'_1, t'_1, ..., k'_m, t'_m)$, in which $k'_1$ points to $t'_1$, $t'_1$ points to $k'_2$,..., $k'_m$ points to $t'_m$ and $t'_m$ points to $w$. $(k'_m, t'_m)$ is called the head of the chain and $(k'_1, t'_1)$ is called the tail of the chain.*

**Lemma 1.** *Consider a graph, in which vertices are kidneys, patients and the waiting list $w$. Every patient points to either a kidney or to $w$. Every kidney points to its paired patient. Then either there exists a cycle, or each pair is the tail of some $w$-chain.*

**Question 3.** Give a proof for Lemma 1.

4.2. **Cycles and Chains Matching Algorithm.** We consider an algorithm that identifies cycles and $w$-chains for performing kidney exchanges. It proceeds by rounds. At every round, patients points to their preferred remaining unassigned kidney or to $w$. When a cycle is detected, the exchange is carried out and involved patients and kidneys are removed from the procedure. If there is no cycle, the algorithm selects a $w$-chain according to some rule. Patients of the selected chain are assigned but the $w$-chain is kept until the end of the procedure. We consider two possible chain selection rules.

- Chain Selection Rule A: Select the longest $w$-chain. In case there are several longest $w$-chains, select the one with the highest priority patient. If the highest priority patient is part of several chains, select the chain with the second highest priority patient, and so on.
- Chain Selection Rule B: Select the $w$-chain starting with the highest priority pair.

**Question 4.** Implement the cycles and chains matching algorithm with rules A and B. Give the pseudo-code of your algorithm. Explain your implementation choices.

4.3. **Example.** We consider the following problem involving 12 donor-patient pairs with the following preferences:

$$t_1 : \; k_9 \succ k_{10} \succ k_1$$

$$t_2 : \; k_{11} \succ k_3 \succ k_5 \succ k_6 \succ k_2$$

$$t_3 : \; k_2 \succ k_4 \succ k_5 \succ k_6 \succ k_7 \succ k_8 \succ w$$

$$t_4 : \; k_5 \succ k_9 \succ k_1 \succ k_8 \succ k_{10} \succ k_3 \succ w$$

$$t_5 : \; k_3 \succ k_7 \succ k_{11} \succ k_4 \succ k_5$$

$$t_6 : \; k_3 \succ k_5 \succ k_8 \succ k_6$$

$$t_7 : \; k_6 \succ k_1 \succ k_3 \succ k_9 \succ k_{10} \succ k_1 \succ w$$

$$t_8 : \; k_6 \succ k_4 \succ k_{11} \succ k_2 \succ k_3 \succ k_8$$

$$t_9 : \; k_3 \succ k_{11} \succ w$$

$$t_{10} : \; k_{11} \succ k_1 \succ k_4 \succ k_5 \succ k_6 \succ k_7 \succ w$$

$$t_{11} : \; k_3 \succ k_6 \succ k_5 \succ k_{11}$$

$$t_{12} : \; k_{11} \succ k_3 \succ k_9 \succ k_8 \succ k_{10} \succ k_{12}$$

We assume that the patients are ranked in a strict priority list based on their index: patient $t_1$ has the highest priority and patient $t_{12}$ the lowest. We adopt the chain selection rule A.

**Question 5.** Illustrate the situation at the beginning of every round and provide the final matching on this example when rule A is chosen.

**Theorem 1.** *Consider a chain selection rule that keeps in the procedure any selected $w$-chain at a non-terminal round. With such a rule, the exchange mechanism is efficient.*

**Question 6.** Give a proof of Theorem 1.

**Question 7.** Show on the example that the exchange mechanism with chain selection rule A is not strategy-proof.

**Theorem 2.** *With the chain selection rule B, the exchange mechanism is strategy-proof.*

**Question 8.** Considering the given example, illustrate the situation at the beginning of every round and provide the final matching on this example when rule B is chosen. Illustrate with a simple example the result of Theoreom 2.

## 5. Integer Programming Formulation

In this section, we consider 0-1 preferences: a kidney is either compatible or not with a given patient and compatible kidneys are indifferent for a patient. We ignore $w$-chains and we consider only cycles of length less than some threshold[5] $K$. Consider a directed graph whose vertices are the kidney-patient pairs. In this graph, there is an edge between pair $i$ and $j$ if the kidney of $i$ is compatible with the patient of $j$. In this graph, any path $\pi = \{i_1, i_2, ..., i_{K+1}\}$ of length $K+1$ is called a *minimal infeasible path.*

**Question 9.** Implement an algorithm that lists all minimal infeasible paths in a directed graph. Give the pseudo-code of your algorithm. Explain your implementation choices. Run your algorithm on test files 'test2.txt' and 'test3.txt', provide the result and the CPU time.

In the considered graph, we look for the maximum number of transplantations that can be performed. If an edge $(i, j)$ is in the solution, we set $x_{ij} = 1$, otherwise $x_{ij} = 0$.

**Question 10.** Formulate an integer linear program (ILP) for this optimization problem.

A classical way of solving ILPs is called Branch-and-Bound (BB). The principle of BB is as follows. We construct a tree of subproblems, whose root corresponds to the initial problem. Each vertex $v$ corresponds to a subproblem, which is generated from its parent in the tree by adding an additional constraint. At node $v$ (a *branch* of the tree), the relaxed subproblem (a linear program) is solved in order to get an upper bound $\bar{z}_v$ of the optimal value for the subproblem. This branch is not further explored if (1) $\bar{z}_v$ is less than a current feasible solution we have already or (2) $\bar{z}_v$ is associated to an integer solution, in which case we can update the current feasible solution or (3) the relaxed subproblem is infeasible.

[5] As transplantations have to be performed simultaneously along a cycle, any cycle length is not possible for practical reasons.

**Question 11.** Implement an algorithm that solves this ILP using Branch-and Bound. You may use an available LP solver at intermediate subproblems. Give the pseudo-code of your algorithm. Explain your implementation choices.

**Question 12.** Run your algorithm on the test files 'test2.txt' and 'test3.txt'. Provide the selected edges, the achieved objective value and the CPU time.

## 6. Quantitative Analysis

We want to quantify the improvement of the cycles and chains mechanism compared to simple approaches.

The main reason for incompatibility is related to the blood type. There are four blood types: A, B, AB, and O. Type O kidneys can be transplanted into any patient; type A kidneys can be transplanted into type A or AB patients; type B kidneys can be transplanted into type B or AB patients; type AB kidneys can only be transplanted in type AB patients. We assume that the frequencies of blood types in the population are the following: type O: 46%, type A: 39%, type B: 11%, type AB: 4%.

We consider $n$ donor-patient pairs whose blood types are distributed according to these frequencies. Compatibility donor sets are built based on the blood type compatibility. In the compatibility set, preferences are drawn randomly. There is a probability 1/2 that a patient with incompatible donor prefer the waiting list to its own donor. Donor-patient pairs are ranked in a priority list according to their index as in the example of the previous section.

We assume that there are $c$ cadaver kidneys available with blood types having the same distribution. At the end of every algorithm, available cadaver kidneys and kidneys offered to the waiting list are assigned to patients in the waiting list starting with the highest priority patient while respecting the compatibility constraint.

**Question 13.** Assume $n = 30$ and $c = 3$ and generate 100 kidney exchange problems. Run the direct donation, the greedy matching and the cycles and chains algorithms on these instances. Comment the results and in particular compare the number of transplantations. What is the effect of running the direct donation algorithm as a preprocessing step to other algorithms or not? What is the gain when allowing multiple-way exchanges instead of 2-way exchanges?