Erasmus University Rotterdam
Erasmus School of Economics
Bachelor Thesis Econometrie en Operationele Research

Are you kidney me?! Multiple lives saved by a single kidney donor

Hester Aarts (510448)

Supervisor:	Dollevoet, TAB
Second assessor:	Cremers, R
Date final version:	9th July 2024

Abstract

In response to the shortage of available kidneys for patients, different countries allow the exchange of kidneys among incompatible patient-donor pairs. These programs are also known as kidney exchange programs. Optimal allocation of kidneys within these programs can be done using integer programming models, but the previously proposed models for this problem seem not suitable for larger scales, due to the large number of constraints and variables. In Constantino, Klimentova, Viana and Rais (2013) a new, more efficient formulation for this problem is constructed and in this paper we compare its performance and results with a previously proposed formulation. Additionally, we try to find a way to convince people to become a new (altruistic) donor, without using a financial aspect since this is not legal in many countries. We do so by estimating the number of lives saved when a new donor applies to the program. We find that the newly constructed formulation is significantly more efficient than the previously proposed. We also find that when a first altruistic donor is added to a Kidney Exchange Program, on average they will save 2.72 lives.

1 Introduction

The waiting list with patients who need a new kidney is long, and the number of donors willing to donate a kidney is still too small (McCormick, Held & Chertow, 2018). Due to an impressive progress in the medical world concerning surgery and medical treatments, and the development of new immunosuppressive drugs, the possibilities of kidney transplants become increasingly advanced (Perico, Ruggenenti, Scalamogna & Remuzzi, 2003). Therefore, the fraction of kidneys transplanted from a living donor instead of a deceased one is increasing the last years. When a patient found a compatible donor willing to donate their kidney, the transplant could be performed.

However, when a patient found a donor that was not compatible due to different blood type, plasma et cetera, the transplant was off the table. This is where the kidney exchange programs come in. When an incompatible patient-donor pair is registered to the program, they can be matched with another incompatible patient-donor pair where the donor from the other pair does have a matching kidney and the other way around. This way two patients can be helped instead of zero. Now the problem arises on how to match these incompatible patient-donor pairs such that as many patients can be helped as possible. This optimisation problem is called the Kidney Exchange Problem (KEP). The KEP is bound by the maximum number of incompatible pairs involved in one kidney exchange plan, which is commonly 2 or 3. It is then known as a 2(3)-way or 2(3)-cycle exchange. These exchanges are illustrated in Figure 1. Generally, when the bound on the size of the cycle increases, more options within the exchanges are possible and more patients can be helped. When the bound does not exist, the KEP becomes an assignment problem and can be solved in polynomial time (Constantino et al., 2013).

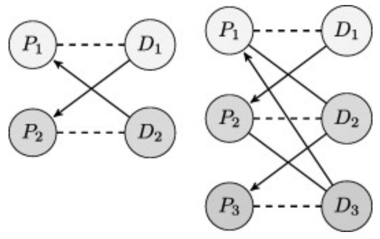


Figure 1: 2-way exchange (left) and a 3-way exchange (right)

Note. Adapted from *New insights on integer-programming models for the kidney exchange problem*, by Miguel Constantino, Xenia Klimentova, Ana Viana, Abdur Rais, 2013, Elsevier. Copyright 2013 by Elsevier B.V.

The restriction on the size of the cycles exists mainly because of logistic issues that arise when a cycle becomes too large. The surgeries need to be done (almost) simultaneously and this becomes unfeasible for a large number. Also, an exchange can be canceled just before surgery due to various medical reasons and then the cycle can not continue. To prevent too many inconveniences, the cycles need to be kept short.

For solving the KEP, two Integer Programming (IP) formulations have been used: the so-called cycle formulation and the edge formulation. Although these formulations work for smaller instances, they are not suitable when the problem becomes of larger scale. The cycle formulation contains an exponential number of variables and the edge formulation an exponential number of constraints. Therefore we compare the new compact extended edge formulation, introduced by Constantino et al. (2013) to improve the performance when the problem scales up, to the old cycle formulation by computational analysis. We also look into the dominance of the new formulation over the old one and obtain upper bounds for optimal solution by using linear relaxations.

Next to comparing the models, we aim to find a way to motivate people to donate their kidney, as there is such a shortage in donors. In many countries it is not legal to provide a financial compensation for donating an organ, and that concept raises many ethical questions as well. Therefore we need to find another way to find more altruistic donors. When looking at the KEP, we see that when we only have patient-donor pairs in the program, the solution can only consist of a certain number of cycles. When introducing an altruistic donor, instead of a cycle a chain can be started (Figure 2). This chain can consist of multiple patients getting the kidney they need, where this would not have happened if there would not have been an altruistic donor.

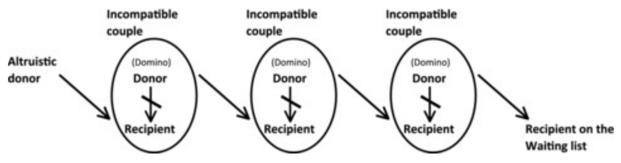


Figure 2: Altruistic donor chain

Note. Adapted from *Altruistic Donor Triggered Domino-Paired Kidney Donation for Unsuccessful Couples from the Kidney-Exchange Program*, by J.I. Roodnat, W. Zuidema, J. van de Wetering, M. de Klerk, R.A.M. Erdman, E.K. Massey, M.T. Hilhorst, J.N.M. Ijzermans, W. Weimar, 2010, Elsevier. Copyright 2010 by Elsevier.

Essentially, an altruistic donor can save not just one life of the patient they donate to, but the lives of all patients in the chain. The potential to save multiple lives rather than just one could be a strong motivation for donors to apply to the program. Therefore we determine what the marginal expected number of lives saved is when an altruistic donor joins the program. We do so by adding one altruistic donor to a given data set and see how many more patients lives can be saved compared to the original problem. By doing this the right number of times, we can find an estimate for the lives saved by adding altruistic donors to the program and add this to a campaign of donor registration. Additionally, we look into the estimates per blood type of the potential donor. This way we could target the population by certain blood types to improve the number of lives saved by the program.

The most important findings of this research are that the EEF seems equally effective but significantly more efficient than the CF. Additionally, the number of lives saved by a newly added altruistic donor to a KEP is on average 2.72.

The outline of this thesis is as follows: in Section 2 an overview is given of all the literature used for this research. Section 3 provides an explanation of the problems solved in this paper. In Section 4 the methodology of the study is explained, which formulations are used to replicate the study of Constantino et al. (2013) and all the (decision) variables of the simulation. Then in Section 5 the results of the studies are displayed. The results of the comparison of the CF and the EEF and the impact of additional altruistic donors on the program are discussed. In Section 6 a conclusion of the studies and their results can be found as well as an idea of how to use them. Finally, in Appendix A an explanation of the used code and how to run it is provided.

2 Literature Review

The Kidney Exchange Problem (KEP) has been extensively studied due to the increasing demand for kidney transplants and the limited supply of donor kidneys. The concept of kidney exchange, initially developed to solve incompatibility issues between patient-donor pairs, has improved due to various mathematical models and optimisation techniques.

Constantino et al. (2013) provides new insights into integer-programming models for KEPs, focusing on improving efficiency and scalability. Their study compared several formulations, including the Extended Edge Formulation (EEF) and the Cycle Formulation (CF). The EEF demonstrated superior performance in handling larger instances of KEPs compared to the other models. The EEF model limits the number of constraints, such that the exponential growth of larger graphs is managed more efficiently.

McCormick et al. (2018) discusses the consequences of the kidney shortage, highlighting the need for innovative solutions like kidney exchange programs. They show the significant impact of kidney exchanges due to the toll of kidney shortages on patients and healthcare systems. Perico et al. (2003) also addressed the donor kidney shortage, promoting better use of available kidneys and the benefits of optimising KEPs to increase transplant rates. De Klerk et al. (2005) provides an extensive description of the existing Dutch Kidney Program, including information on its origin, logistics and practicalities.

The role of altruistic donors, or Non-Directed Donors (NDDs), has been important in improving the effectiveness of KEPs. Studies such as those by De Klerk et al. (2010) analyses the optimal chain length for kidney paired exchanges, finding that longer chains started by NDDs can significantly increase the number of successful transplants. This finding shows the importance of including NDDs in KEPs to maximise transplant rates. Another study by De Klerk, Witvliet, Haase-Kromwijk, Claas and Weimar (2008) discusses the challenges and successes of national living donor kidney exchange programs, further showing the effectiveness of including NDDs in these programs.

Additionally, the work by Cecka (2010) introduced the Calculated Panel Reactive Antibody (CPRA) as a new measure of sensitisation for transplant candidates. CPRA helps in identifying compatible donor-patient pairs more accurately, thus improving the matching process in KEPs. Understanding blood group compatibility, as outlined by Sanquin (n.d.), is also important in the matching procedure.

The optimisation of KEPs involves complex algorithms and models. Trimble (2019) provides an example using Python and Gurobi to solve kidney exchange instances by a practical approach to implement these mathematical models. His work highlights the importance of computational tools in optimising kidney exchanges and making them more accessible to the public.

Finally, Cornelio, Furian, Nicolò and Rossi (2019) provides a study conducted on the impact of adding deceased donors to a KEP. This is an alternative way to improve the outcome of a KEP and could be looked upon for improving donor recruitment in a further research.

In conclusion, the literature on KEP highlights the evolution of mathematical models and optimisation techniques used to solve the kidney shortage crisis. From the development of advanced integer-programming models like the EEF to the addition of altruistic donors and even the combining of living and deceased donor programs, significant steps have been made in improving the effectiveness of matching procedures for kidney exchanges. However, the need for innovative ideas to increase donor registration and optimise the current use of available kidneys remains an interesting area of research.

3 Problem description

The data used is provided by Erasmus School of Economics, a department of Erasmus University Rotterdam. The data includes 4 differently sized kidney exchange pools with 10 instances each and $n = \#$pairs and $m = \#$altruistic donors. The sizes are S ($n + m = 50$), M ($n + m = 70$), L ($n + m = 100$) and XL ($n + m = 200$). These instances include a compatibility matrix with either a 1 for compatibility or a 0 for incompatibility between the donor of a pair or an altruistic donor and a patient of another pair. We introduce the directed graph G where the vertices are donor pairs or altruistic donors and the edges represent the compatibility between the donor of a pair (or an altruistic donor) and the patient of another pair. The edge is directed from the donor the kidney could come from to the patient it could be donated to. An altruistic donor in graph G does not have a corresponding patient and therefore only has outgoing edges (Constantino et al., 2013). The goal is now to find the cycles and chains such that the number of patients helped is maximised, while adhering to the constraints.

4 Methodology

For the comparison of the formulations we run the cycle formulation and the extended edge formulation. First, we provide an explanation of the formulations (Constantino et al., 2013).

4.1 Cycle formulation

The first integer programming model is the cycle formulation. Let $C(k)$ be the set of all cycles in G with length at most k. A cycle is an ordered set of arcs and a variable z_c is defined for every cycle $c \in C(k)$:

$$z_c = \begin{cases} 1 & \text{if cycle } c \text{ is selected for the exchange,} \\ 0 & \text{otherwise.} \end{cases}$$

Now, $V(c) \subseteq V$ is the set of vertices in cycle c. With $w_c = \sum_{(i,j)\in c} w_{ij}$, the model is written as follows:

$$\begin{aligned} \text{Maximise:} \quad & \sum_{c\in C(k)} w_c z_c & & & (1) \\ \text{Subject to:} \quad & \sum_{c:i\in V(c)} z_c & \leq 1 \qquad & \forall i \in V & (2) \\ & z_c & \in \{0,1\} \qquad & \forall c \in C(k) & (3) \end{aligned}$$

In this paper, we only use the weights of either 0 or 1, since we do not add values to the performed transplants. The transplant either successful or unsuccessful. Thus, we maximise the number of transplants using the objective function (1). The constraints (2) ensure that every donor can only donate and every patient can only receive one kidney. The number of cycles can grow exponentially with k.

4.2 Extended edge formulation

The new and improved integer programming model is the extended edge formulation (EEF), which is expected to be more efficient than the cycle formulation. For the EEF, we take L copies of the graph G and to each assign an index l, such that L is an upper bound on the number of cycles in a solution. For every copy l , the maximum number of arcs in a cycle equals k and every node $i \in V$ can be used in at most one cycle. For formulating the model we will be using the variables x_{ij}^l:

$$x_{ij}^l = \begin{cases} 1 & \text{if arc } (i,j) \text{ is selected to be in copy } l \text{ of the graph,} \\ 0 & \text{otherwise.} \end{cases}$$

Now, the extended edge formulation is as follows:

$$\begin{aligned}
\text{Maximise:}\quad & \sum_{l\in\{1,...,L\}} \sum_{(i,j)\in A} w_{ij} x_{ij}^l && && (4)\\
\text{Subject to:}\quad & \sum_{j:(j,i)\in A} x_{ji}^l &&= \sum_{j:(i,j)\in A} x_{ij}^l && \forall i \in V, \quad \forall l \in \{1,...,L\} && (5)\\
& \sum_{l} \sum_{j:(i,j)\in A} x_{ij}^l &&\leq 1 && \forall i \in V && (6)\\
& \sum_{(i,j)\in A} x_{ij}^l &&\leq k && \forall l \in \{1,...,L\} && (7)\\
& x_{ij}^l &&\in \{0,1\} && \forall (i,j) \in A, \quad \forall l \in \{1,...,L\} && (8)
\end{aligned}$$

The objective function (4) will be maximised by maximising the total weight of the arcs taken from all copies of the graph, and thus the total number of kidneys donated. The first set of constraints (5) ensures that in each copy l the number of kidneys donated by donor i is equal to the numbers of kidneys received by patient i (either both 0 or both 1). The second set of constraints (6) states that a node can only be selected in at most one copy of the graph, such that a donor/patient only intervenes once. The last set of constraints (7) ensures that at most k edges can be used from each copy of the graph. Since now each copy of the graph only allows cycles of length k or less, it prevents cycles to become larger than k.

4.3 Adding altruistic donors

For adding the altruistic donors, we need to adjust the formulations. Exchanges where altruistic donors are involved are called domino paired chains (DPC). With the altruistic donors, the problem is still presented as a weighted directed graph but consisting of $n+m$ nodes instead of n. Here, $V = 1, ..., n+m$, with n being the number of incompatible donor pairs and m the number of altruistic donors. Thus, nodes $1, ..., m$ correspond to the altruistic donors and $m+1, ...n+m$ to the incompatible pairs. For each altruistic donor a dummy patient is matched that is compatible with all altruistic donors $j \in 1, ..., n$.

Now, we set k' to be the maximum length of a DPC. We can let the set of all cycles in G with a maximum length of k including only incompatible pairs, and all cycles in G including an altruistic donor with a maximum length of k' be denoted by $C(k, k')$. Now the cycle formulation can be directly performed on $C(k, k')$ instead of $C(k)$.

In the other formulation, the main impact concerns the cardinality constraints that determine the maximum size of the cycle. These constraints now need to be divided into two sets: one for cycles that include an altruistic donor, and another for cycles that include only incompatible pairs.

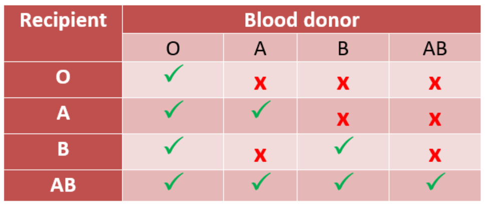
For the EEF, constraints (7) are split into two new sets of constraints:

$$\sum_{(i,j)\in A: i,j\in\{l\}\cup\{m+1,\ldots,L\}} x_{ij}^{l} \leq k' \qquad \forall l \in 1,\ldots,m \tag{9}$$

$$\sum_{(i,j)\in A: i,j\geq m+1} x_{ij}^{l} \leq k \qquad \forall l \in m+1,\ldots,L \tag{10}$$

4.4 Simulation

For the following part of the study a simulation is made where a number of random altruistic donors is added to the program. First, the already existing altruistic donors in the program are removed and then the new donors are added. For every added donor, a row is added to the compatibility matrix containing ones and zeros according to the compatibility of the kidney of the added altruistic donor with the patient of the pair. The blood type has to match according to Figure 3.



Recipient	Blood donor			
	O	A	B	AB
O	✓	x	x	x
A	✓	✓	x	x
B	✓	x	✓	x
AB	✓	✓	✓	✓

Figure 3: Blood type matching

Note. Adapted from *The ABCs of ABO Blood Types*, by Amanda Maxwell, 2016, Canadian Blood Services. Copyright 2024 by Canadian Blood Services.

The calculated panel-reactive antibody (cPRA) also plays its part in the compatibility of kidneys of donors and patients. The cPRA provides a percentage of organ donors that will be incompatible for a candidate (Cecka, 2010). Therefore, the compatibility of a random altruistic donor is determined by matching blood types and is then 0 or 1 with the probability of the cPRA that it is 0.

Then, for every experiment, every blood type and every chain capacity, 1 to 10 random altruistic donors are added, in 10 simulations. So for example, in experiment 0 to 9 we add 1 random altruistic donor with blood type A, repeat this 10 times, run the code for chain capacity 3 and take the average total score over the simulations and experiments. The marginal increase in total score is analysed and displayed as the first data point in the plot Blood Type A in Figure 5 in the k=3 line. For the base line the problem is also solved for chain capacity 0, where the added altruistic donors have no effect on the total score since they can not donate to any patient.

This method results in $\#bloodtypes \times \#experiments \times \#simulations \times \#donors \times \#chaincapacities = 4 \times 10 \times 10 \times 10 \times 5 = 20,000$ instances. These are all solved using the EEF formulation since this is the most efficient and fast formulation according to the results found when replicating Constantino et al. (2013). In the Dutch KEP, the median number of couples participating in a match run is 47 (range 16-66) (De Klerk et al., 2008). Therefore, for the simulation the program in the given data where $n + m = 70$ (size M) is chosen to use for solving the problems. Since this study is especially about the altruistic donors and thus the chains, the maximum length of the cycles is fixed. For this fixed value, 3 is chosen because according to De Klerk et al. (2005) it is the median of the cycle length in the Dutch KEP, where the cycles have no maximum length for research purposes and therefore the lengths range from 2 to 12. According to De Klerk et al. (2010) the optimal chain length in the Dutch program is also 3.

Since the next part of the study is on the donors and the chains, and the goal is to see how many lives the donors can save and thus how many patients fit in the added chains, we prefer to not put a too strict maximum on the chain length, but still keep the length realistic. Therefore, the chain capacity varies from 3 to 6. For the maximum number of donors added, we looked at the results of the marginal increase in total score when the donors are added and see that when the number of donors approximates 10, the program seems to become saturated and the marginal increase approaches 0. Thus adding more donors beyond this point does not provide additional value.

4.5 Computation

Experiments were conducted to compare the proposed and existing formulations regarding the time required to find an optimal solution and the LP gaps relative to the upper bounds of the models' linear relaxations. Using Gurobi 11.0.2 and Python 3.10.14, CPU times and bounds were measured on a computer with an 11th Gen Intel(R) processor running at 3.00 GHz, 16 GB of RAM, and Windows 10 Pro 22H2.

5 Results

The results of the studies provide a comparison of the different integer programming formulations for the Kidney Exchange Problem (KEP) and the impact of adding altruistic donors to the kidney exchange program.

5.1 Formulation comparison

Since later on we will study the impact of altruistic donors, for the comparison of the formulations the altruistic donors are also included. For running these formulations, the Python code of Trimble (2019) is adjusted and used. The provided data is formatted to the used input data in the original code such that it can be used directly. The pairs and altruistic donors are split up in different files and the numbering is changed to the natural numbers from 0 to $n + m - 1$, where the first numbers correspond to the altruistic donors and the rest with the pairs (e.g. $\{0, ..., m-1\}$ correspond to the altruistic donors and $\{m, ..., n+m-1\}$ correspond to the pairs).

The compatibility matrix is converted to a data frame with in each row a donor and a patient (vertices in the eventual graph) and a 1 (edge) representing the compatibility between the two. Then the cycle and chain capacities are determined. Since the reasons for limiting the size of cycles (logistical constraints, the risk of donors withdrawing from the program, and medical considerations) would similarly apply to chain size, we chose to keep the cycles and chains the same maximum size (k). To give a good review of the formulations, we chose 4 realistic cycle and chain capacities, namely k = 3, 4, 5 and 6.

Then, for every size {S, M, L, XL} and every capacity {3, 4, 5, 6}, every experiment {0, ... , 9} is run. To analyse the performance of the formulations the means of the number of variables, number of constraints, total runtime and eventual score are calculated and presented in Table 1. The performance of the cycle formulation (CF) and extended edge formulation (EEF) for different problem sizes (S, M, L, XL) and cycle capacities (3, 4, 5, 6) are summarised.

- Effectiveness: As we can see, the total scores of the CF and the EEF are nearly the same, suggesting that both the formulations are equally effective. However, when the problem becomes too large, the CF performs quite poorly. When $n+m = 100$ and $k = 6$, experiment 1, 7 and 9 caused an out of memory error. Therefore, the results of those experiments are excluded and the total score of the CF differs from the total score of the EEF (41.57 against 43.9). For the instances where $n + m = 200$ and $k \geq 5$, the same error is observed by all experiments. The results of those instances are therefore not obtained and are missing from Table 1. For all the other instances, the CF's and the EEF's performance is equally effective.

- Variables and constraints: As expected, the number of variables in the CF increases exponentially when the problem becomes larger. The number of constraints in the EEF increases as well but maintains a more manageable number of variables and constraints compared to CF, especially for larger problem sizes.

- Efficiency: the EEF performs significantly more efficient than the CF. Where the runtime of the EEF only becomes noticeably larger than 1 second where $n + m = 200$ and $k \geq 4$, the CF's performance already deteriorates in terms of efficiency in all instances where $k = 6$ and $n + m \geq 100$.

The findings from this study highlight the superior performance of the EEF over the CF in handling larger instances of the KEP. The EEF's ability to manage a large number of variables and constraints efficiently makes it a suitable choice for large-scale kidney exchange programs, and therefore for the study of finding the impact of additional altruistic donors on the KEP, since 20,000 instances need to be solved for this purpose.

Table 1: Results of the comparison between the CF and the EEF

$n+m$	k	**CF**				**EEF**			
		#var	#con	runtime	score	#var	#con	runtime	score
50	3	857.8	38.3	0.029930	17.3	457.7	295.1	0.030986	17.3
	4	4,509	38.8	0.085719	18.2	956.9	484.2	0.049639	18.2
	5	22,902.8	38.8	0.433687	18.4	1,546.3	648	0.060446	18.4
	6	114,744.8	38.9	2.412568	18.4	2,131.6	767.9	0.080232	18.4
70	3	3,258.6	53.5	0.077858	27.9	917.5	550.2	0.047217	27.9
	4	24,283.8	54.1	0.529918	28.9	2,360.5	1,104	0.115430	28.9
	5	181,362.4	54.1	4.191112	28.9	4,335.1	1,468.6	0.191382	28.9
	6	1,366,101.4	54.2	38.785833	28.9	6,140.4	1,696.7	0.229896	28.9
100	3	20,299.3	85.6	0.445806	42.9	2,889.9	1,253.3	0.125143	42.9
	4	217,500.7	86.1	5.800198	43.8	7,734.8	2,949.8	0.359990	43.8
	5	2,238,093.7	86.3	61.575942	43.9	16,474.6	4,208.4	1.063376	43.9
	6	16,180,765	85.29	583.785293	41.57	25,055.7	4,848.8	1.327288	43.9
200	3	192,331.6	178.7	4.780431	93.4	12,311.6	4,747	0.668592	93.4
	4	4,045,987.2	179.4	132.177121	94.5	48,926.5	13,966.3	5.642355	94.5
	5	-	-	-	-	130,119	19,340.4	22.201730	94.6
	6	-	-	-	-	202,600.8	21,519	45.615155	94.7

5.2 Impact of additional altruistic donors

This study examines the impact of adding altruistic donors to the kidney exchange program. The number of simulations is looked upon. First, we started off with 100 simulations per instance, meaning that 100 times a fixed number of random altruistic donors with a certain blood type is added to the program consisting of only incompatible pairs. In each simulation, the only difference is the added compatibility rows consisting of ones and zeros that represent the possibility for the added donor to donate their kidney to the patient of the pair. When running the 100 simulations, we noticed that every simulation had the exact same total score and it seemed that the simulations provided different outcomes but always the same total score. To test this hypothesis, we solved 1000 simulations where one donor is added to the program where

$n + m = 200$ (XL), experiment 0. The cycle capacity is fixed on 3 and the chain capacity on 10 since the length of the chain can impact the total score the most. An altruistic donor with blood type O is added to the program since a donor with this blood type impacts the total score the most as well. When the 1000 simulations are solved, we notice that the total score is 998 times 106 and only 2 times differs, namely 105. The convergence and the (non-visible) 95% confidence intervals are displayed in Figure 4. We conclude that increasing the number of simulations does not add value to the average total score and therefore choose to do 10 simulations per instance.

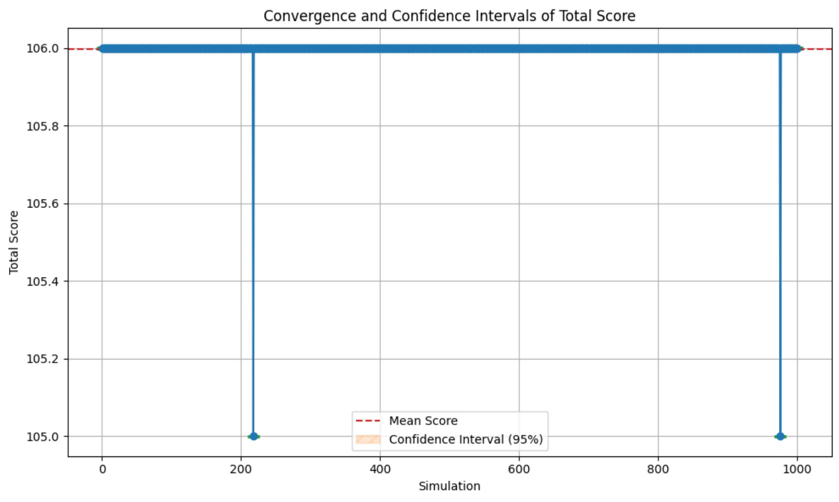


Figure 4: Convergence and confidence intervals of 1000 simulations

The simulations added 1 to 10 altruistic donors to the program and evaluated the marginal increase in the total score, representing the total number of possible transplants. Figure 5 presents the results for different blood types (A, B, AB, O) and chain capacities (3, 4, 5, 6). On the x-axis the number of added altruistic donors is presented and on the y-axis the average marginal increase in total score (thus extra lives saved by this donor). For every chain capacity a different line is displayed.

When analysing the results, we see that adding altruistic donors consistently increases the total number of possible transplants. Starting with adding a first altruistic donor, on average 2.72 lives are saved. As more donors are added, the marginal increase in total score diminishes. This indicates a saturation point where the added donors contribute less to the total score. The blood type of the additional donor impacts the total number of possible transplants significantly. Donors with blood type AB contribute the least (0.22 for the first donor) and donors with blood type O contribute the most (3.59 for the first donor). This is in line with expectations since individuals with blood type O are universal donors and individuals with blood type AB are universal recipients. The contribution of donors with blood type A and B seems quite similar.

Looking at the chain capacities, we see that higher chain capacities generally lead to a higher total score, as longer chains allow more patients to be matched. This is in line with expectations as well because with longer chains the added donor can make a bigger impact. The results show that the optimal chain length for maximizing the number of transplants in this study is between

3 and 5. This matches with the optimal chain length found by De Klerk et al. (2010).

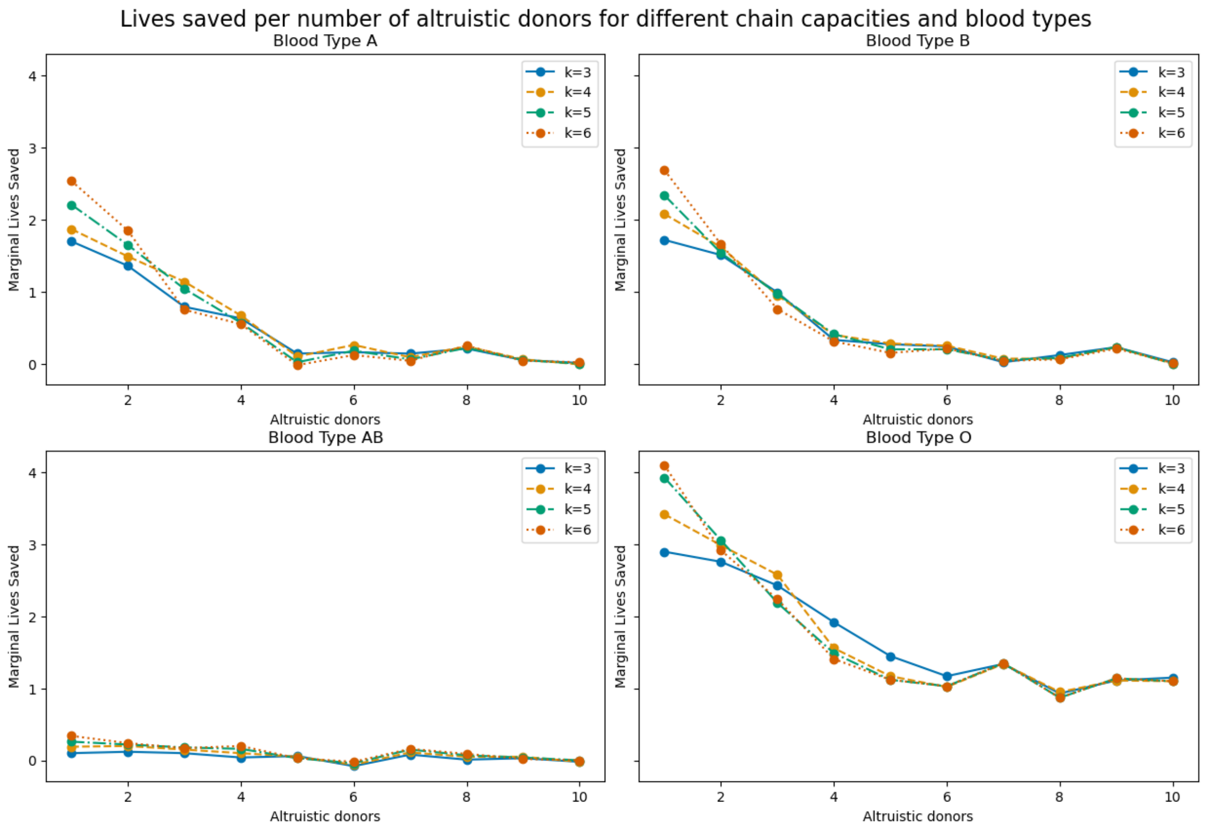


Figure 5: Impact of additional altruistic donors to the KEP

In Table 2 the average lives saved for the first # of added donors is displayed. The first added donor with blood type A can save 2.08 lives on average. The first 3 donors with blood type A can each save 1.15 lives on average, et cetera.

Table 2: Average # lives saved by first added donors

	First # of additional donors			
Blood types	1	3	5	10
A	2.08	1.15	0.83	0.49
B	2.21	1.19	0.84	0.50
AB	0.22	0.13	0.11	0.07
O	3.59	2.11	1.65	1.16

To find the general number of lives saved by a random additional donor independent from their blood type, we look at the distribution of blood types in the Netherlands. According to the Dutch national blood bank Sanquin (n.d.), 43% of the Dutch people has blood type A, 9% blood type B, 3% has blood type AB and 45% blood type O. With this distribution, the average number of lives saved for a random first added donor is calculated by (11) and equals 2.72.

$$43\% \times 2.08 + 9\% \times 2.21 + 3\% \times 0.22 + 45\% \times 3.59 = 2.72 \tag{11}$$

6 Conclusion

In this paper, the previously proposed cycle formulation is compared to the new and more efficient extended edge formulation. After this, the impact of adding altruistic donors to a kidney exchange program is studied.

The findings from replicating the study of Constantino et al. (2013) highlight the superior performance of the Extended Edge Formulation (EEF) over the Cycle Formulation (CF) in handling larger instances of the Kidney Exchange Problem (KEP). The effectiveness of the formulations seems to be quite similar, but the number of variables and constraints is better kept under control by the EEF. Also, the efficiency of the EEF compared to that of the the CF is significantly higher. The EEF's ability to manage a large number of variables and constraints efficiently makes it a suitable choice for large-scale kidney exchange programs and therefore for the following part of the study.

The findings of studying the impact of additional altruistic donors provide valuable insights into the benefits of donors in a kidney exchange program. The results suggest that even a small number of altruistic donors can significantly increase the number of successful transplants, especially when the donors have blood type O. For a random first donor added to the program, independent of their blood type, the expected number of lives saved is 2.72. This information can be used to design targeted campaigns to recruit altruistic donors, emphasizing the potential to save multiple lives.

Moreover, the diminishing marginal returns with additional altruistic donors indicate a need for a balanced approach in donor recruitment. While more altruistic donors are beneficial, the focus should also be on improving the efficiency and logistics of kidney exchange programs to maximise the impact of each donor. For the Dutch Kidney Exchange Program, the Dutch Transplant Foundation performs a computer match procedure every 3 months (De Klerk et al., 2008). From our results we see that the impact of the first (few) donors added to the program is more significant than that of the last donors, due to saturation of the program. This could mean that it would be helpful to spread the altruistic donors over the multiple match procedures every year. Adding 3 donors every procedure could have a bigger impact on the number of patients helped than adding 10 to the first run of the year, and having none left for the maintaining procedures. This could be looked upon in further research.

Also, an earlier study is conducted on adding deceased donors to the kidney exchange program to start a chain, as well as an altruistic donor (Cornelio et al., 2019). This could mean that deceased donors also save multiple lives and the donor recruitment could focus on this as well.

Overall, these findings contribute to the current efforts to optimise kidney exchange programs, offering inventive ideas for improving donor recruitment and program efficiency.

References

Cecka, J. (2010). Calculated pra (cpra): the new measure of sensitization for transplant candidates. *American Journal of Transplantation*, *10*(1), 26–29.

Constantino, M., Klimentova, X., Viana, A. & Rais, A. (2013). New insights on integer-programming models for the kidney exchange problem. *European Journal of Operational Research*, *231*(1), 57–68.

Cornelio, C., Furian, L., Nicolò, A. & Rossi, F. (2019). Using deceased-donor kidneys to initiate chains of living donor kidney paired donations: Algorithm and experimentation. In *Proceedings of the 2019 aaai/acm conference on ai, ethics, and society* (pp. 477–483).

De Klerk, M., Keizer, K. M., Claas, F. H., Witvliet, M., Haase-Kromwijk, B. J. & Weimar, W. (2005). The dutch national living donor kidney exchange program. *American Journal of Transplantation*, *5*(9), 2302–05.

De Klerk, M., Van Der Deijl, W. M., Witvliet, M. D., Haase-Kromwijk, B. J., Claas, F. H. & Weimar, W. (2010). The optimal chain length for kidney paired exchanges: an analysis of the dutch program. *Transplant International*, *23*(11), 1120–1125.

De Klerk, M., Witvliet, M. D., Haase-Kromwijk, B. J., Claas, F. H. & Weimar, W. (2008). Hurdles, barriers, and successes of a national living donor kidney exchange program. *Transplantation*, *86*(12), 1749–1753.

McCormick, F., Held, P. J. & Chertow, G. M. (2018). *The terrible toll of the kidney shortage* (Vol. 29) (No. 12). LWW.

Perico, N., Ruggenenti, P., Scalamogna, M. & Remuzzi, G. (2003). Tackling the shortage of donor kidneys: how to use the best that we have. *American Journal of Nephrology*, *23*(4), 245–259.

Sanquin. (n.d.). *Bloedgroepen.* `https://www.sanquin.nl/over-bloed/bloedgroepen`. (Accessed: 2024-06-15)

Trimble, J. (2019). *Solve kidney-exchange instances using python and gurobi.* `https://github.com/jamestrimble/kidney_solver`. GitHub.

A Programming code

For the programming of this study, we use the code of Trimble (2019). We adjust the data provided by the Erasmus University of Rotterdam so that is fits the code. For the replication we run kidney_solver-master/real_data/edit.py. This turns the Excel files into separate .input and .ndds files. Then we run kidney_solver-master/kidney_solver/main.py where we adjust the parameter sets to our liking and remove the simulation for-loops. Make sure that the input and output files are correct, not with _extension. In kidney_solver-master/kidney_solver/main.py and in kidney_solver-master/kidney_solver/kidney_solver.py we manually adjust the size. So for the replication we do this 4 times, once for every size.

Then for the extension, we first run kidney_solver-master/extension_data/edit_extension.py for a certain blood type, manually adjusted. Then again we run kidney_solver-master/kidney_solver/main.py, with the correct input and output files. When running the extension, make sure to turn $cap\_cha = cap\_cyc$ off in both the for loops (and on for the replication). We do this 4 times, for every blood type once, manually adjusted. Then to generate the plots we run kidney_solver-master/kidney_solver/plots.py. When running the code, both for the replication and the extension, it is important to make sure that the solution files of the previous runs are deleted, since they are not automatically overwritten and will append to the old results.