BACHELOR

The Fairness Level in Failure-Aware Kidney Exchange

Buist, Merel B.C.

Award date:
2020

Department of Mathematics

www.tue.nl

Author
Merel Buist (1253409)

Supervisor
B.M.L. Smeulders

Date
July 3, 2020

The Fairness Level in Failure-Aware Kidney Exchange

Bachelor Final Project

Merel Buist (1253409)
m.b.c.buist@student.tue.nl

Abstract

Kidney exchange programs are keen to improvements. The first programs have yet started in the late 90's, which clarifies the room for development. The hypothesis of failure-aware matching leading to more transplants is explored in this paper, as well as the influence the failure-aware method has on the fairness of the solution. Through integer linear programming we solve several constructed models. We find that failure-aware matching does lead to more transplants, with an increase of approximately 42.5 percent. However, the new method also creates more inequality between different types of patients. By manipulating the modeled success probabilities such that the difference between patients becomes the same as in the deterministic method, the solution becomes more fair. Fair solutions nonetheless decrease outcomes; The most fair solution decreases the amount of transplants by about 6.5 percent compared to the failure-aware method. It is left to the reader to answer the ethical question of which balance between fairness and efficiency to choose. The application of recourse to the solution results in an even larger increase of transplants, which makes it interesting to research fairness with recourse applied.

Technische Universiteit
Eindhoven
University of Technology

Table of contents

Title
The Fairness Level in Failure-Aware Kidney Exchange

Where innovation starts

Technische Universiteit
Eindhoven
University of Technology

Table of contents

Title
The Fairness Level in Failure-Aware Kidney Exchange

Where innovation starts

I. Introduction

Worldwide there are hundreds of thousands of people waiting for a kidney. Even though the Netherlands is the country with relatively the largest amount of kidney transplants through living donors (Biró et al., 2019), on the 1st of June, 2020, there were still 814 patients waiting for a kidney in the Netherlands only (Nederlandse Transplantatie Stichting, 2020). Because of the shortage in supply of kidneys, people die waiting on a kidney. In some cases, a patient will find a family member, friend or other acquaintance which is willing to donate his or her kidney to them. However, it is quite common that this donor, due to a difference in blood type or other immunological discrepancies, cannot donate to that particular patient. For this problem the well-known kidney exchange program has been set up. The patient and donor sign up for such a program, listed as a patient-donor pair. The program matches these kind of patient-donor pairs to each other, such that the enrolled patients still receive a kidney. The donor of a pair donates to the patient of another pair, whose donor in turn also donates to another patient. A trivial example is imagine we have patient-donor pairs (P_A, D_A) and (P_B, D_B). Donor A cannot donate to patient A, but can donate to patient B. Vice versa donor B is incompatible with patient B, but compatible with patient A. This program could then match these two pairs, such that donor A donates to patient B and donor B donates to patient A. See Figure 1 for a visualization of this situation.

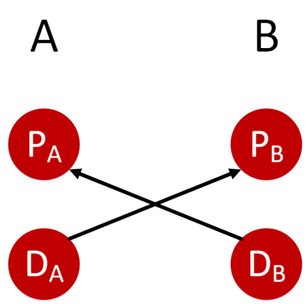

Figure 1: Matching Two Pairs

This example is a 2-cycle between A and B, however larger cycles are also possible of course. The program could also make a chain, which begins with a non-directed donor donating a kidney to a patient-donor pair and then goes on with these pairs donating to one another. A non-directed donor is a donor which is not enrolled together with a patient in the program. He or she donates a kidney, however does not want one back in return. Therefore, it is possible to form a chain. A chain can be endlessly long.

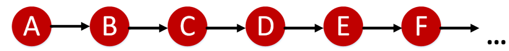

Figure 2: Example of a chain

Because of these programs, substantially more people receive a kidney. In 2017 for instance, worldwide 32,990 patients received a kidney from a living donor (Global Observatory on Donation and Transplantation, 2017). These programs try to maximize the number of transplants, by planning as many transplants as possible. In mathematical terms, the program maximizes the number of arcs in the solution. Thus, we have a directed graph (V, A), in which V is the set of patient-donor pairs and non-directed donors, and A is the set of all possible transplants. So if a donor, either non-directed or in a patient-donor pair, is compatible with another patient, then there is an arc in the graph directed from that donor to that patient. We refer to the method of maximizing the planned amount of transplants as the deterministic method in this paper.

Nevertheless, most of the times this maximization does not lead to the maximum number of transplants eventually being executed. Planned transplants still have a probability of failing, due to positive crossmatch, illness, or other causes (Dickerson, Procaccia, & Sandholm, 2019). If one failure occurs, the whole cycle in which the failing arc or vertex is located will be canceled, or the chain will end at the failure. The cycle canceling is due to the domino effect that if one patient-donor pair does not receive a kidney they will not donate one as well, which causes no one to donate nor receive a kidney. A chain ending at failure can be explained by a simple illustration; Looking at Figure 3, one can see that if the third arc fails, D won't receive a kidney and therefore not donate one to E. This goes on, causing all transplants after C to be canceled. The transplants which were planned earlier however do go through, as all patient-donor pairs got their kidney so also donate one.

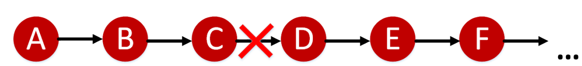

Figure 3: Chain Failure

Consequently, a new method of matching pairs has arisen: the method of failure-aware kidney exchange. In this approach, not the amount of planned transplants is maximized, but the expected amount of transplants is maximized. Hence, failure probabilities are taken into account and accordingly cycles and chains are made. Doing so can significantly change results, a small example can illustrate this; Suppose we only consider arc failure, and the arcs have a probability of failure of 70 percent. By not taking failure into account one might just form a large cycle, say of size 6. However, if only one arc fails the whole cycle fails. By forming two 3-cycles, one arc failing will only cause one of the cycles to fail. Forming three 2-cycles results in an even smaller loss of transplants, as one arc failing only causes one of the three cycles to fail. The expected amount of transplants for all cases is the success probability of a cycle multiplied by its length, summed for all cycles. This gives the outcomes:

$$\textbf{One 6-cycle} = 0.3^6 \cdot 6 = 0.004374$$
$$\textbf{Two 3-cycles} = 0.3^3 \cdot 3 + 0.3^3 \cdot 3 = 0.162$$
$$\textbf{Three 2-cycles} = 0.3^2 \cdot 2 + 0.3^2 \cdot 2 + 0.3^2 \cdot 2 = 0.54$$

Hence, failure-aware matching makes huge differences in outcomes, which makes it interesting to research.

II. Problem Description

In this paper, only cycles will be considered. Because of this, there are no non-directed donors in the graph. The optimization of this kidney exchange program can be written as an integer linear program. There are various ways of formulating this, in this paper we do as follows:

$$\text{maximize} \quad \sum_{c \in C} w_c x_c \tag{1}$$
$$\text{subject to:} \quad \sum_{c: i \in c} x_c \leq 1 \qquad \forall i \in V \tag{2}$$
$$x_c \in \{0,1\} \qquad \forall c \in C \tag{3}$$

(Abraham, Blum, & Sandholm, 2007)
The set of all vertices in graph G is V, and the set of all possible cycles in the graph G is called C. Each cycle has its own weight, w_c. In the deterministic case, the case of maximizing the planned amount of transplants, the weight of each arc is equal to 1 and hence w_c = $|c|$ = the number of arcs in cycle c. In the failure-aware exchange, w_c equals the *expected* amount of transplants of the cycle. x_c is a binary variable indicating whether the cycle is contained in the solution or not, 1 indicating it is, 0 indicating it is not. (2) ensures that each vertex is contained in at most one cycle. This can be explained by the fact that each patient-donor pair can receive at most one kidney and donate at most one kidney.

There is however a pitfall to failure-aware optimizing, namely the fairness of the solution. In this problem, each patient has a calculated Panel Reactive Antibody (PRA) associated to them. This PRA can take a value between 0 and 100. The number is determined through the compatibility between the patient and other donors, considering they are blood type compatible. The more likely it is that the patient will be incompatible with another donor, the higher the PRA of a patient (Working Group 2, 2020). So for example, if a person is incompatible with everyone, it will have a PRA of 100. Patients with a high PRA are characterized as highly sensitized patients; They have a small probability of being compatible with another patient-donor pair, and therefore have a lot less arcs directed at them in the graph of all possible transplants. This causes highly sensitized patients to have a smaller probability of being matched in the eventual solution. The formal definition we use for highly sensitized patients is the following:

A patient is highly sensitized if and only if it has a PRA bigger or equal to 80.

Due to this failure-aware matching, people with a low success probability of arcs are also expected to have a lower chance of being matched. Commonly, highly-sensitized patients-donor pairs also have a lower success probability of the arcs (Glorie & K.M., 2012). In this paper, the fairness of this is considered and tackled. How much less likely are these pairs to be matched? In what way should the model be adjusted to combat this unfairness? What effect does this have on the number of expected transplants matched?

This paper will begin by reviewing previous knowledge obtained on the subject. After that, it will continue by explaining the different models and how the data was created for these models. Next, both the deterministic and non-deterministic model will be implemented and an analysis will be performed on both models and their differences in effectiveness. Following, the fairness in the optimal solutions will be explored. With this information, it will be determined how to accordingly adapt the model such that the optimal solution is fair for everyone. Various cases between the fair and non-fair, i.e. non-deterministic without compensation for fairness, will be analyzed. Then, the consequences of these adjustments per case will be evaluated. We also implement the non-deterministic case with recourse, to observe the influence of this method. Lastly, the methods in this paper will be concluded and discussed, giving recommendations for further research.

III. Literature Research

Nowadays, kidney exchange has evolved to a quite common health-care surgery. It saves the lives of people suffering from kidney failure. In 2017, there were 90,306 kidney transplants across the entire world. Moreover, 36.5 percent of these transplants involved a living donor (Global Observatory on Donation and Transplantation, 2017). Due to the fact that this surgery has such a big success across the world, the relevance of the subject and the complications accompanied to this kidney exchange program, many scientists have already researched this subject. The main complication in this program is the lack in 'supply' of kidneys and the big probability of a transplant between donor and patient failing. This last problem is combatted by making the most effective solutions possible, which is what scientists try to research. In this paragraph, we will summarize their findings.

A. Deterministic Model

The first basic model for solving this problem is the deterministic model. As stated earlier, the weight w_c is equal to 1 for each arc. Through the integer linear program (1) the problem is solved. One method of solving is simply enumerating all cycles and chains with their properties. Another method is applying column generation such that the cycles and chains used for the integer program are wisely chosen, instead of using them all. For large data pools this will speed up the process, however for small ones we have seen that this may not be the case (Dickerson, Manlove, Plaut, Sandholm, & Trimble, 2016). Both methods nevertheless need a maximum on the length of a cycle, say n. In a majority of the KEPs, this maximum is set at 3 or 4 (Dickerson et al., 2019) (Biró et al., 2019).

The kidney exchange problem in deterministic setting, with the maximum length of a cycle greater than 2 but still finite, is NP-hard (Abraham et al., 2007).

B. Failure-Aware Model

Through further investigation, a revised model to this first one has been constructed: The failure-aware model. To illustrate the significance of failure-aware matching: In the American UNOS program, 93 percent of the planned matches failed (Dickerson et al., 2019), in the UK failure-aware NHS program, this was only 37 percent in the period between 2014 and 2018 (NHS, 2018). The problem can be solved failure-aware through multiple ways.

There are papers using stochastic optimization and papers using robust optimization. The most substantial difference between the two is that stochastic optimization finds a solution with the highest expected outcome of transplants, which requires explicitly modeling the failure probabilities, and robust optimization finds the best 'worst case scenario'. In other words, robust optimization finds a solution that given a set of possible scenarios, the lowest outcome these scenarios can give is optimized, where optimal usually means the highest amount of transplants taking place. An advantage of this robust method is the fact that there is no need to approximate the probability of failure for arcs or vertices, which is a pitfall in the stochastic approach. As that probability is modeled and all results rely on that probability in the stochastic method, one might question the accurateness of the results, considering that the failure probability might be incorrect.

1) Robust Optimization

In such a robust optimization, an uncertainty set is constructed for a parameter which is hence uncertain as the name says. This set then exists of the values the parameter can possibly take. To illustrate this, we use an example. Consider S the set of all scenarios s in which in total X arcs fail. Take the binary vector q_s, which indicates which cycles fail and which do not given scenario s. This means that if $q_{s,c}$ is 0, the cycle c fails in scenario s. Let $|c|$ be the size of cycle c, i.e. the number of transplants in c, and M is the set of all possible solutions, so each possible combination of cycles such that each vertex is used at most once. Let x_c be a binary variable being equal to 1 if and only if cycle c is in the solution M, therefore x is a vector indicating for each cycle whether it is in the solution or not. Lastly, C is the set of all possible cycles c in the graph. The solution that is found by applying robust optimization is:

$$\max_{x \in M} \min_{s \in S} \sum_{c \in C} q_{s,c} \cdot |c| \cdot x_c$$

The worst case scenarios in the uncertainty set S are taken, i.e. the minimal outcomes of the sum for each solution x. For all these minimal outcomes which each correspond to an x, robust optimization finds the solution x which has the highest minimal outcome, i.e. the highest amount of transplants. Therefore, it optimizes the worst possible outcome the final solution can have. (McElfresh, Bidkhori, & Dickerson, 2019)

2) Stochastic Optimization

The stochastic approach instead optimizes the expected number of transplants resulting from a solution. This method has more risk to it, as the worst case outcome is usually lower than for the robust method. Nevertheless, in expectation the number of transplants will be at least as high, and likely higher, as the robust method. This approach is the one that this paper will use.

To perform stochastic optimization one requires the failure probabilities p. These failure probabilities can be determined by various methods. First of all, by making a guess based on the given data one can determine a failure probability which is equivalent for all arcs. This is a quite simple method. The weight of a cycle equals the size of a cycle multiplied by the success probability. A previous research has estimated this constant failure probability to be equal to 70 percent by analyzing the data they got from the UNOS, an American Kidney Exchange program (Dickerson et al., 2019). This same paper also uses two other manners of modelling failure probabilities. The first is with a bimodal distribution; 25 percent of the possible transplants have a low failure rate, which is determined by taking a number from a uniform distribution between 0.0 and 0.2. The other 75 percent of the possible transplants have a high failure rate, where now a uniform distribution between 0.8 and 1.0 is used. The mean failure rate is then still 70 percent. The second other manner is by taking a normal distribution with mean 0.70 and standard deviation σ and drawing the failure probability for each single arc from this. Notice that σ has not been initialized. The paper systematically changes the standard deviation σ, observing the results for each different value. One could also pick failure probabilities by drawing from a uniform distribution between 0 and 1, as done in (Klimentova, Pedroso, & Viana, 2016)

Using this probability of failure, thus failure aware optimizing, we have seen that better results are found than by applying the maximum cardinality optimizing, i.e. the deterministic method of maximizing the planned amount of arcs in the final solution. We see this for instance in the papers of (Dickerson et al., 2019) and (Maria, Carmelo, Taniar, Apduhan, & Hutchison, 2014).

3) Recourse

A failure can either be an arc or a vertex failure. When applying recourse to a solution it is important to distinguish between these two types of failures. Recourse is the concept of allowing the possibility of rearranging the final solution once the failures for this solution have been determined. For instance, look at Figure 4. The illustration shows a 3-cycle with an embedded 2-cycle. Suppose the matching makes the 3-cycle, and after the failures have been determined one observes that the arc between A and B fails. Then, through recourse, a new cycle can be chosen, namely a 2-cycle consisting of A and C. In this way, one failure does not mean that the whole cycle necessarily ceases, some patients might still get a kidney.

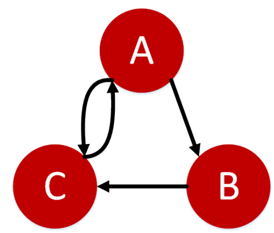

Figure 4: A 3-cycle with an embedded 2-cycle

Due to this opportunity, the total amount of transplants is again increased. There are different forms of recourse, the paper (Klimentova et al., 2016) applies three different forms. Firstly, there is the option of 'no recourse', which speaks for itself. Notice that this is the same as the previously explained model, where a solution is determined and nothing is adjusted. The second is 'internal-recourse', which means that there can be rearranged within the cycles found in the solution themselves. The example explained above forming the 2-cycle A-C after failure in Figure 4 is an example of internal recourse. The third form is 'subset-recourse', which enables the possibility to add patient-donor pairs who initially were not in the solution to the new solution. For each cycle, there is a subset of patient-donor pairs which are strongly connected[1]. The program finds a solution where the cycles and their corresponding subsets are all disjoint from one another. Then, if an arc or vertex fails in the cycle, a new cycle or new cycles can be formed with the subset to obtain the new solution. Depending whether a vertex or arc fails, new rearrangements can be made.

[1]The definition of strongly connected is that all vertices in the strongly connected set can reach one another.

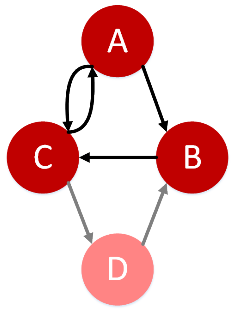

Figure 5: Two 3-cycles intertwined

Figure 5 illustrates this last concept. Suppose the program formed the 3-cycle A-B-C again, however after the failures have been determined we detect that vertex A fails. D is a vertex not yet included in the entire solution *and* is strongly connected with the 3-cycle, so is in the subset of the cycle for subset-recourse. Instead of the whole cycle failing, the 3-cycle B-C-D can be formed.
To make this an efficient process and not let the time period of solving explode, the size of a subset is limited to a certain number.

IV. Creation of the Data

The model requires a data pool full of patient-donor pairs with their corresponding characteristics. Saidman et al. 2006, describes a method of how this data can be created. Each pair has a blood type belonging to the patient and a blood type belonging to the donor. The same applies for gender. The patient also has a certain PRA. Furthermore, the patient and donor must be incompatible with each other to participate in the kidney exchange program. Distributions for blood type, gender and PRA are obtained from Saidman et al. 2006. With these distributions characteristics are assigned to patients and donors. Per PRA, the paper also gives the probability of a patient being compatible with a donor. The paper says that women have a higher positive crossmatch, i.e. not being compatible, probability with their partner. In fact, the adjusted PRA then equals 100-0.75(100-PRA). To determine whether the patient and donor in a pair are compatible or not, blood type compatibility is determined and if they are blood-compatible then it is determined whether the two have a negative crossmatch or not. Only if they are incompatible, will they be added to the pool. (Saidman, Roth, Sönmez, Ünver, & Delmonico, 2006)

Once this pool of patient-donor pairs is created, the directed graph is constructed. Hence, the arcs between the vertices must be established. By comparing all patient-donor pairs with each other and checking for compatibility, the arcs are created. If the patient of pair A is compatible with the donor of pair B, then an arc directed from B to A is added. Similar as to how compatibility between a patient and donor in one pair internally was determined, the compatibility between a patient and donor of different pairs is determined.

Another prerequisite for the failure-aware model is the data of the failure probabilities. The study of Glorie, 2012, analyzed data from Dutch KEPs. They found a distribution for the failure probability of arcs depending on the PRA level of a patient. For low PRA's, i.e. lower than 10, the failure probability was only 6 percent. For PRA's between 10 and 80, the ingoing arcs of the patients had a failure probability of 31 percent. For high PRA's, i.e. above 80, there was a failure probability of 44 percent. Therefore, by looking at the PRA of the patient located at the end of the arc, the success probability of that arc is determined.

V. Mathematical Model

As explained earlier, this paper will have a stochastic approach to the kidney exchange program optimization. Considering certain failure probabilities, the highest expected amount of transplants can be reached. We predict that there will eventually be more transplants through this stochastic approach than by using the deterministic approach. In this section we will go further into the models used to obtain our results.

The model will be solved by the technique of enumeration and not column generation, which is adequate taking in mind the size of the data pool used for this research.

A. Creation of Cycles

First, we will explain how the cycles are formed. The length of the cycles has been limited to the size 4. Many KEP's use 3 as limit (Dickerson et al., 2019), however to more clearly investigate the effect of deterministic or failure-aware matching we have chosen the length 4. The higher one chooses their limit, the worse the outcome of the deterministic case will be. Higher limits nevertheless also cause for rapid increases in running time, thus this will decrease the efficiency of the solver. Hence, 4 has been chosen as the limit.

By checking the arcs between vertices for possible

cycles the set of enumerated cycles can be made. There either should be a closed walk after two vertices, three vertices or four vertices, i.e. cycle size two, three or four respectively.
The precise formation of the cycles will be explained with the help of pseudocode, see Algoritm 1. It begins by creating the set C in which all cycles will be stored. Then, it will search through all patient-donor pairs in the data pool. For an outgoing arc a_1 of the patient-donor pair it is looking at, i_1, it checks all cycles which this arc can make and adds them to the set of cycles C. Once all possible cycles containing a_1 have been added, the arc is removed from the set of outgoing arcs A_1. Removing this arc ensures that no identical cycles are added twice. The reason for this is that when a new patient-donor pair i_1 is taken and one of the outgoing arcs of i_1 goes back to an earlier checked patient-donor pair i_x, the updated set of outgoing arcs for i_x will be empty.

For arc a_1, the algorithm looks at which pair i_2 it is directed to, and goes through all outgoing arcs of i_2. If the arc taken, so a_2 in the algorithm, directs back to the first patient-donor pair i_1, a 2-cycle is added to C. When adding a cycle to the set C we immediately enumerate the cycle. If the arc a_2 directs to another patient-donor pair i_3, the algorithm looks if bigger cycles can be formed with the outgoing arcs of i_3. Similarly, the algorithm checks whether the arc a_3 forms a 3-cycle or not. If not it checks for a 4-cycle. If a 4-cycle can be formed with arc a_4 the cycle is added to set C and the while loop is ended, as the maximum cycle size is 4. Otherwise, the arc is removed from the temporary set to check if another arc does form a 4-cycle. Once all these arcs are checked or if a 3-cycle was formed the arc a_3 is removed to check if another arc may form a 3-cycle or a 4-cycle. This step is also done for a_2, to check if another arc may form a 2-cycle or larger cycles. Lastly, when all cycles for a_1 have been checked and added, a_1 will be removed and a new outgoing arc of i_1 will be taken. Once all outgoing arcs of patient-donor pair i_1 have been checked and thus A_1 is empty, the algorithm goes to the next patient-donor pair and repeats the whole process.

B. Weights of a Cycle With No Recourse Applied

Firstly, we focus ourselves on the computation of the weight of a cycle. As mentioned before, the weight of a cycle is its success probability multiplied with the length of the cycle, given that no recourse is applied. We distinguish between the two different methods, deterministic and failure-aware. By doing so, results can show which manner is the best, thus induces the highest amount of transplants taking place.

1) The Deterministic Case

Here, the success probability is plainly equal to 1, as this case does not consider the possibility of arcs failing. It is not failure-aware, but simply maximizes the total amount of arcs in the final solution. Therefore, the weight of cycle c is equal to its length, i.e. $|c|$.

2) The Failure-Aware Case

Using the success probabilities found in the paper of Glorie, 2012, all the arcs in the graph are enumerated with their corresponding success probability. With these probabilities the weights can easily be determined. We use the notation a_i for arc i with success probability q_i. Define A as the set containing all arcs in cycle c. Hence, the weight of cycle c is

$$|c| \cdot \prod_{i:a_i \in A} q_i$$

C. Weights of a Cycle With Internal Recourse Applied

This weight determination is always failure-aware, as recourse assumes there to be failures which effects can then be restrained by rearranging the solution. Applying recourse to a solution changes the weight of a cycle. Namely, when one arc fails, it does not necessarily imply that the whole cycle fails anymore. The new weight is calculated by considering all possible events of different arcs failing, computing the deterministic solution for the succeeding arcs, and then summing the product of the solution with the probability of the event. Thus, if we have cycle c with x arcs between all vertices which all form at least one internal cycle of c, the amount of events with different arcs failing is 2^x. Let s_i be the solution, i.e. the amount of transplants planned, for event i. The probability of event i happening is p_i, which is computed by multiplying the success probabilities of the succeeding arcs with the fail probabilities of the failing arcs in event i. The weight of cycle c then is:

$$w_c = \sum_{i=1}^{2^x} s_i \cdot p_i$$

Instead of checking for all events in the cycle, there is a shortcut. For instance, in Figure 4 if the arc between B and C fails and also the arc between C and A, one already knows there are no cycles possible anymore. There is no need to check whether the other arcs fail or not, as the solution remains the same. Hence, in practice

Algorithm 1 Cycle Formation

```
Create empty set of cycles C
for All patient-donor pairs i1 do
    Get an arc a1 ∈ A1, the set of outgoing arcs of i1
    while A1 is nonempty do
        Get patient-donor pair i2 which is at the receiving end of arc a1
        Make temporary set T2 = A2, where A2 is the updated set of outgoing arcs of i2
        Get an arc a2 ∈ T2
        while T2 is nonempty do
            Get patient-donor pair i3 which is at the receiving end of arc a2
            if i1 = i3 then
                Add 2-cycle with i1, i2, a1 and a2 to C
            else
                Make temporary set T3 = A3, where A3 is the updated set of outgoing arcs of i3
                Get an arc a3 ∈ T3
                while T3 is nonempty do
                    Get patient-donor pair i4 which is at the receiving end of arc a3
                    if i1 = i4 then
                        Add 3-cycle with i1, i2, i3, a1, a2 and a3 to C
                    else
                        Make temporary set T4 = A4, where A4 is the updated set of outgoing arcs of i4
                        Get an arc a4 ∈ T4
                        while T4 is nonempty do
                            Get patient-donor pair i5 which is at the receiving end of arc a4
                            if i1 = i5 AND i4 != i2 then
                                Add 4-cycle with i1, i2, i3, i4, a1, a2, a3 and a4 to C
                                break
                            end
                            Remove arc a4 from T4
                            Get new arc a4 ∈ T4
                        end
                    end
                    Remove arc a3 from T3
                    Get new arc a3 ∈ T3
                end
            end
            Remove arc a2 from T2
            Get new arc a2 ∈ T2
        end
        Remove arc a1 from A1
        Get new arc a1 ∈ A1
    end
end
```

we do not compute all different events separately. The code for this is based on the code used in the paper of Klimentova et al. 2016, with an addition. This addition being the method of choosing arcs. Instead of simply taking an arc in the set, the algorithm now cleverly chooses an arc, such that it may end the process earlier if possible. We will explain the process by the help of the pseudocode given in Algorithm 2.

The weights are determined cycle by cycle. A cycle is taken, and it is determined whether it has length 2 or larger. If the length is 2 there can be no internal cycles or arcs so the whole process is skipped and the cycle immediately gets the same weight as with the failure-aware case without recourse. If the length is bigger, then first the sets are created with all internal cycles and all internal arcs that form an internal cycle. A cycle c2 is considered internal for c if all vertices in c2 are in cycle c. By going through all possible cycles c2 in the graph, we check if it is an internal cycle of c, and if so the whole cycle c2 is added to the set of cycles, and the arcs of c2 are added to the set of

Algorithm 2 Assigning the Weights Given Internal Recourse

```
Create vector weightsIntRec to store the weight per cycle
for All possible cycles c in the graph do
    if Cycle c has length bigger than 2 then
        Create the set ArcsInternal
        Create the set CyclesInternal
        Add all arcs of c to ArcsInternal
        Add cycle c to CyclesInternal
        for All possible cycles c2 in the graph do
            if c contains c2 and c2 != c then
                Add all arcs of c2 to ArcsInternal
                Add cycle c2 to CyclesInternal
            end
        end
        Sort CyclesInternal from large to small cycles
        weightsIntRec[ID of c] = ComputeRecWeight(ArcsInternal, CyclesInternal, 1)
    else
        weightsIntRec[ID of c] = failure-aware weight for cycle c
    end
end
```

arcs if they are not yet in that set. c itself and its arcs are also added to these sets. Once this has been done, the weight of cycle c is determined by the function ComputeRecWeight.

The function ComputeRecWeight is a recursive function. If there are no cycles left in the set of cycles, then the solution is definitely 0 so it returns 0. The same for if the success probability of that event is 0, then the event will not happen so we return 0. If the set of cycles only contains one cycle, then that is the only solution left. There is no need to test other arcs whether they fail or succeed. The probability of the event happening and the probability of the yet undetermined arcs in that cycle succeeding is multiplied by the length of the cycle. Another situation is that all arcs have been determined and hence the set *ArcsInternal* is empty. In that case the function solves the Integer Linear Program 1 and returns this solution multiplied with the probability of the event occurring.

If none of these situations have yet occurred, the function takes the first arc a in the first cycle c of *CyclesInternal* that has not yet been determined to have failed or succeeded. The set *CyclesInternal* is ordered by length, with the largest cycle first and the smallest last. Therefore, if it has been determined that cycle c succeeds, there is no need to check whether other cycles with the same patient-donor pairs succeed, as cycle c will be at least as large as all other cycles that can still succeed. The order of cycles with the same length does not matter, as the solution remains the same regardless of their order. If there are no arcs of cycle c in *ArcsInternal*, i.e. check=0, that means the cycle has succeeded as it was still in the set *CyclesInternal*. The set *newCyclesInternal* consists of only cycles which are disjoint from the succeeding cycle. The set *newArcsInternal* consists of the arcs of the cycles in *newCyclesInternal* without the ones which have already been determined to succeed. The arcs which have been determined to fail are have been excluded earlier, as a cycle containing a failed arc is not in the set *CyclesInternal*. The solution equals the probability multiplied by the length of cycle c, plus the recursive function given the new sets starting with the same probability. In the other situation, if there is still at least one arc to check for cycle c, i.e. check=1, then the algorithm considers two situations: Either arc a fails or succeeds. For both situations the recursive function is executed. The final outcome is the solution of both cases summed. The solution that is then returned as the weight of the cycle c is the summation of all outcomes of possible events multiplied by their probabilities of occurring. This weight is the expected outcome of the cycle.

The programming solver that is used to solve this optimization problem is Gurobi. With the integer linear program given in Section II, the Problem Description, and inputting the corresponding weights for a method the Gurobi model finds the solution. The programming language where the code is written in is Java.

```
Function ComputeRecWeight(ArcsInternal, CyclesInternal, sucprob)
    if CyclesInternal is empty or sucprob = 0 then
        return 0.0
    end
    if CyclesInternal contains only 1 cycle c then
        Get arcs from c which are in set ArcsInternal
        sucprob2 = 1.0 · the success probabilities of those arcs
        return sucprob · sucprob2 · length of c
    end
    if ArcsInternal is empty then
        Solve the ILP 1 with the cycles in CyclesInternal and the deterministic weights of the cycles
        sol = outcome of ILP
        return sucprob · sol
    end
    Initialize arc a
    Get first cycle c from CyclesInternal
    Create int check = 0
    for all arcs a2 in cycle c do
        if ArcsInternal contains arc a2 then
            a = a2
            check = 1
            break
        end
    end
    if check = 0 then
        LB = sucprob · size of c
        newCyclesInternal = CyclesInternal
        for all cycles c2 in CyclesInternal do
            if cycle c and cycle c2 are not disjoint then
                remove cycle c2 from newCyclesInternal
            end
        end
        newArcsInternal = ArcsInternal
        for all cycles c2 in newCyclesInternal do
            for all arcs a2 in cycle c2 do
                if ArcsInternal contains arc a2 AND newArcsInternal does not contain arc a2 then
                    Add a2 to newArcsInternal
                end
            end
        end
        sol = LB + ComputeRecWeight(newArcsInternal, newCyclesInternalsuc, sucprob)
        return sol
    else
        CyclesInternalsuc = CyclesInternal
        CyclesInternalfail = CyclesInternal without the cycles that contain arc a
        ArcsInternal = ArcsInternal without arc a
        sucprobsuc = sucprob · success probability of a
        sucprobfail = sucprob · failure probability of a
        solsuc = ComputeRecWeight(ArcsInternal, CyclesInternalsuc, sucprobsuc)
        solfail = ComputeRecWeight(ArcsInternal, CyclesInternalfail, sucprobfail)
        return solsuc + solfail
    end
End Function
```

VI. Results from the No Recourse Model

As the weights have now been determined, we can obtain our first results. These will reveal the differences in the deterministic case and the failure-aware case. In this section we will observe and analyse these differences.

The performance of both methods can be seen below. These numbers are based on 50 instances with 100 patient-donor pairs per instance.

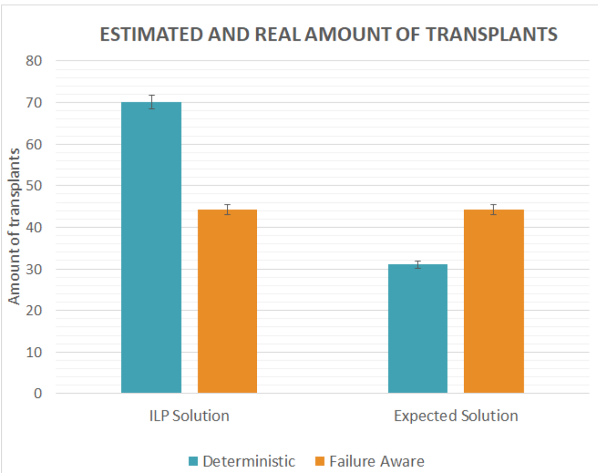

Figure 6: Solutions for the Deterministic and Failure-Aware case

The 95 percent confidence intervals can also be seen in the figure, which shows the accurateness of the numbers. All confidence intervals in this paper are computed by multiplying $z_{0.05} = 1.96$ with the square-root of the variance of all data divided by the amount of different instances. This number is the halfwidth, so:

$$\text{halfwidth} = 1.96 \cdot \sqrt{\frac{\text{variance data}}{\text{amount of data values}}}$$

By adding and subtracting this number from the mean of the data, one gets the 95 percent confidence interval of the data.

The left two bars in Figure 6 present the solution given by the integer linear program. We see that the deterministic case gives a significantly higher amount than the failure-aware case. The deterministic case does not keep in mind the option of failure of cycles when obtaining the ILP solution, which explains this outcome. From the two right bars, which represent the expected amount of transplants deriving from the solution after failure, we see that the failure-aware case gives a higher result, approximately 42.5 percent higher. We confirm the prediction that failure-aware matching leads to more matches than non failure-aware. Moreover, in Figure 6 one can see that the ILP solution for the failure-aware method is exactly the same value as the expected solution, which is correct as the ILP amount represents the expected amount for the failure-aware case. For the deterministic method this definitely not true, the ILP amount is much higher than the expected amount as the failures have not been accounted for in the ILP solution.

Figure 7 visualizes the difference between the expected outcome for the deterministic case and for the failure-aware case per run. The orange bars represent the expected amount of extra transplants the failure-aware solution results in. 50 instances were tested, so all with different data pools meaning new patient-donors pairs were created for each of the 50 runs. For all 50 instances, the failure-aware outcome is higher than the deterministic outcome. We confirm that overall the failure-aware case leads to more transplants.

To strengthen the conclusion of the failure-aware case leading to a higher solution even more, a paired t-test has been done on the data. The test compares all expected outcomes of the deterministic case to the failure-aware case. These are 50 values, as 50 different instances were run, and there were a 100 patient-donor pairs per instance. The test takes as null hypothesis that the mean difference between the two values is zero. The alternative hypothesis that is tested is that the difference in means is not equal to zero. The p-value of the test is smaller than $2.2 \cdot 10^{-16}$, which means the null hypothesis is rejected. Thus, the mean difference between the two is not equal to zero. It is in fact equal to 12.79076 with the 95 percent confidence interval being [12.13071, 13.45080]. In Figure 8 one can see the difference between the two. For each instance, there is an expected solution for both the deterministic and failure-aware method, and one can see the correlation between the two by the line that is drawn between them. A lot of the lines have a similar slope, meaning that in a lot of the instances the increase due to failure-aware matching is the same. This can be confirmed by the low standard deviation of only 2.3 transplants.

Looking at the distribution of the cycle length per method we can verify whether our presumption of the failure aware method using cycles with smaller lengths than the deterministic method was correct. Observing Figure 9, this is indeed the case. By running over 50 different instances with 100 patient-donor pairs, these results were obtained. The figure shows the mean amount of cycles with a certain length used in the solution of a method for one instance. The

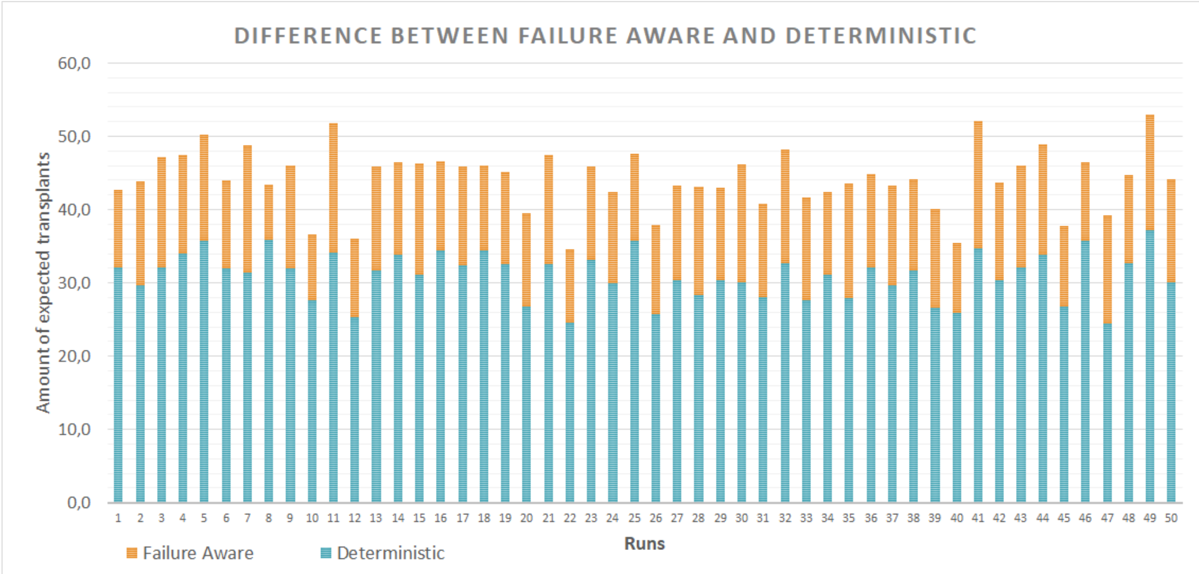

Figure 7: Difference between expected outcome of the Deterministic and Failure-Aware case per run

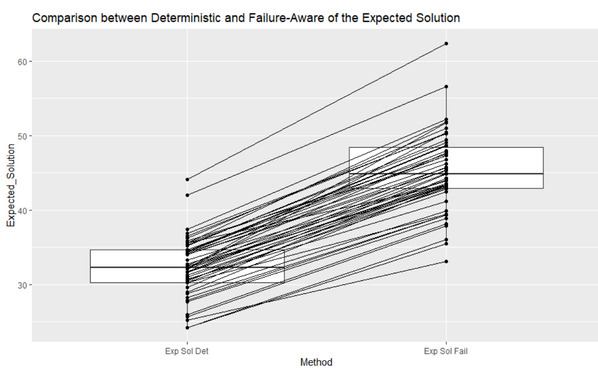

Figure 8: Difference between the expected solution of the Deterministic case compared to the Failure-Aware case

deterministic method does not bear in mind the failure probability of larger cycles. There are substantially more 4-cycles possible in the graph of patient-donor pairs than there are 3- or 2-cycles possible, which results in the deterministic case having relatively a lot more 4-cycles in the solution than other cycle lengths. The failure-aware method does bear in mind this failure probability, thus causing there to be no cycles of length 4, and mostly cycles of length 2.

VII. Fairness

As we know from literature, the highly sensitized patients have on average a higher failure probability, and patients with a low sensitization level have a much

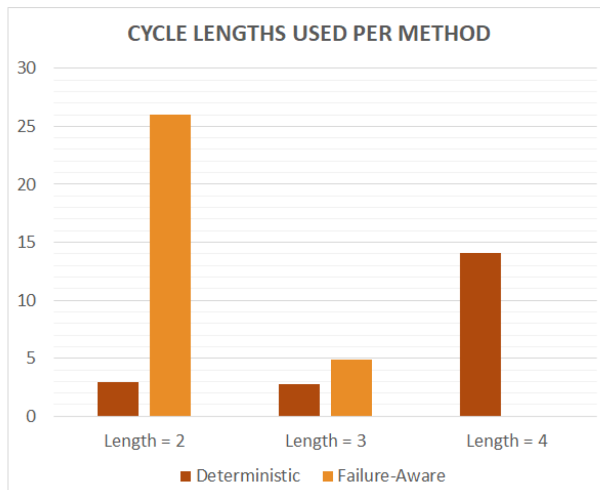

Figure 9: Distribution of the cycle lengths per method

lower failure probability. In this section we investigate the differences in chances of being matched created through failure-aware matching. We try to make it as fair as possible. Lastly, we look at the influence this adjustment has on the final amount of people being matched, and discuss how favorable this outcome is.

We begin by defining three different groups: One with patients having a low sensitization level, i.e. PRA lower than 10, one with patients having normal sensitization level, i.e. PRA between 10 and 80, and the other one with highly sensitized patients, i.e. PRA higher than 80. We look at the expected probabilities for all different groups to be matched and to receive a kidney. Hence, if there is a cycle in the solution containing them, it must also succeed. These probabilities can be seen in Figure 10.

All results in this section are obtained by testing over 20 different instances with 100 patient-donor pairs.

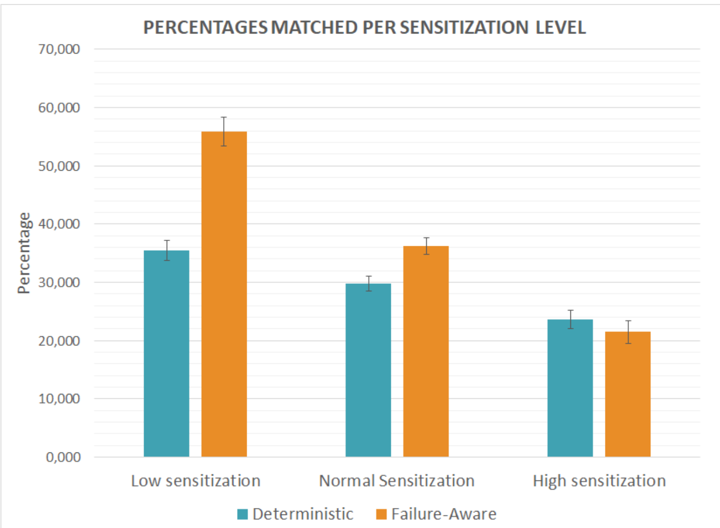

Figure 10: Per sensitization level the expected probability of pairs receiving a kidney for Deterministic and Failure-Aware case

For the deterministic case, the probabilities of a pair to be matched for each sensitization level approach one another much more than for the failure-aware case. The lower the sensitization level, the slightly higher the probability. This is caused by the fact that the lower one's PRA, the more incoming arcs a pair has from other pairs in the graph, hence the easier they are to match. Also, once they are matched, low sensitization patients have a higher chance of their transplant succeeding.
For the failure-aware case, one can observe that percentage-wise, much more patients with a low sensitization level are matched compared to the deterministic case. Similar for the normal sensitization level, failure-aware matching leads to a higher probability of them receiving a kidney. Only for the high sensitization level does that probability decrease. This is as we predicted, because these patients have a fairly high chance of arc failure compared to the two other levels. The differences between all three levels themselves are increased, which only makes the solution more unfair.
To tackle this unfairness, we will adjust the modeled success probabilities of arcs. Before we do this, we need to define what we consider fair.

A. What is Fair?

The definition of fair in this paper is:

The solution is called 'completely fair' if and only if the ratio between the probabilities of a pair to be matched per sensitization level stays the same as for the deterministic case.

So, if the chance of being matched for the low sensitization level patients increases by 5 percent compared to the chance for the deterministic case, then the chances for the two other groups should also increase by 5 percent. One could argue that the solution is already fair if none of the groups have a lower chance of being matched due to failure-aware matching. Through a simple example we will illustrate why we do not consider this fair though choose the definition stated above.

Suppose we have the probabilities for the deterministic case, the same as in Figure 10. Furthermore, the probabilities of being matched per sensitization level for the failure-aware case are the same for normal and high sensitization, but for low sensitization the probability rises to 90 percent, see Figure 11.

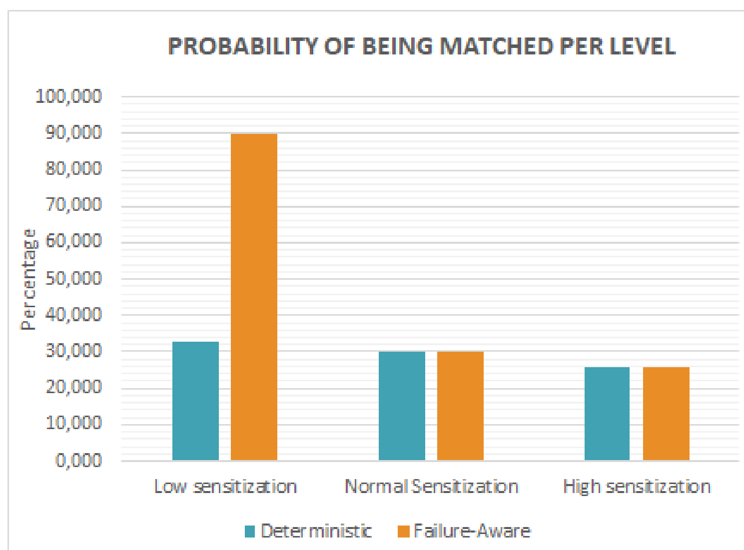

Figure 11: An example of the influence of failure-aware matching on a sensitization level's chance to be matched

The argument of a solution being fair if no groups decreases in probability would deem this distribution to be fair. The definition in this paper does not do so, as the ratios changed substantially, see Figure 12. The figure shows the ratios between the probabilities of being matched per sensitization level group. Thus, if the low sensitization level has probability x of being matched, the normal level has probability y and the high sensitization level has probability z, then the ratio for the low sensitization group is:

$$\frac{x}{x+y+z}$$

As x makes up for a much larger proportion of the total in the failure-aware method in this case, the ratio for the low sensitization group is a lot larger. As the total amount has increased but neither the normal sensitization amount nor the high sensitization amount has increased, those ratios both decrease for the failure-aware method.

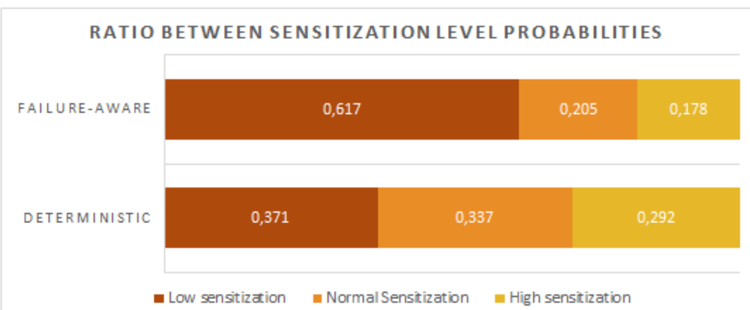

Figure 12: An example of the influence of failure-aware matching in the ratio between probabilities of being matched per sensitization level

The newly innovated method only causes an increase for the low sensitization level, while for the other two levels the probability stays the same. One would think that in time through innovations everyone is benefited, and we create a better healthcare system for everyone. However, in this example this is only the case for a particular group. Because of the immunological characteristics of a person, their position compared to others becomes worse relatively. A bigger difference between groups of people is created. Low sensitization patients will be less likely to die waiting on a kidney, and can live healthy again significantly faster than other sensitization level patients in this case. Hence, we do not deem it to be fair only if no probability decreases, as the difference between levels can be, significantly, increased. We seek that improved methods for kidney exchange programs benefit everyone equally.

By simply choosing a failure probability which is equal for all three groups, one would expect the influence of failure-aware matching on the ratios to vanish. The reason for this is that the new method will not prefer lower sensitization level patients over higher sensitization level patients. By choosing the mean failure probability as constant failure probability, we obtained the following results from again 50 different instances and 100 patient-donor pairs per instance:

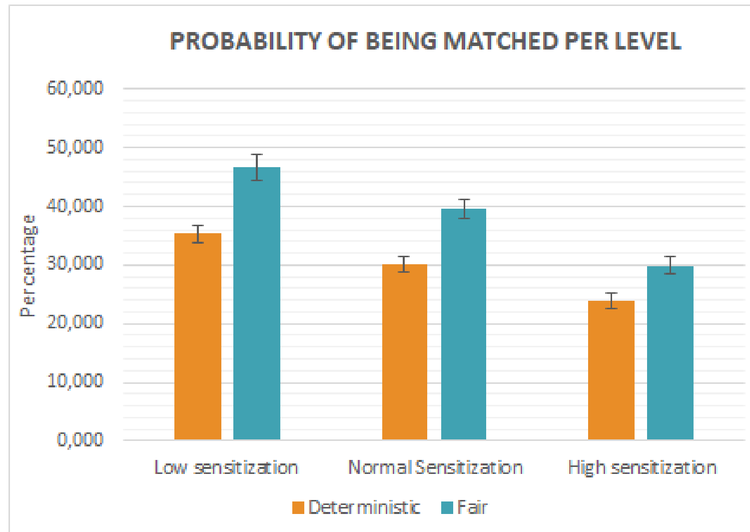

Figure 13: Probabilities to be matched per group for the fair and deterministic method

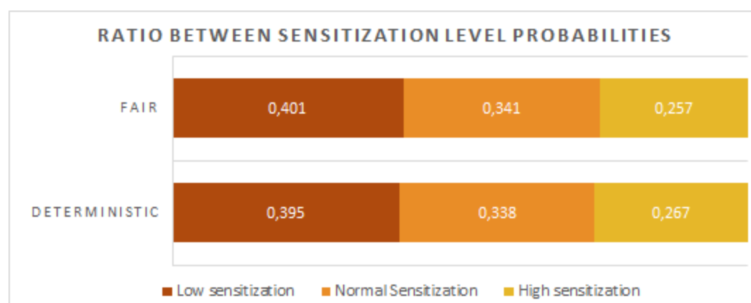

Figure 14: Ratios between probabilities of being matched per sensitization level for the fair and deterministic method

The expected amount of transplants is determined with the cycles in the solution and the success probabilities of Glorie, 2012. One can clearly see in Figure 14 that the ratios for both methods are similar. Therefore, this method is considered fair. Figure 13 visualizes the probabilities of being matched per group, where one can observe that all probabilities are increased. To be more specific, relatively speaking they all increase equally.

As the amount of transplants is affected by these fair probabilities compared to the failure-aware method without adjusting, we also want to look at methods with probabilities in between. Observing how fair they are and how much they affect the eventual solution, one can choose which method is optimal in their opinion.

B. Designing Fair Methods

Previously, the probability of an arc to succeed was equal to the real success probability of the arc, thus the expectation of the arc. Now, we will manipulate these probabilities for the model in such a way that the solution contains patients from different sensitization levels in the same ratio as for the deterministic case. We define the new success probabilities as:

$$q_{new} = \begin{cases} (1+\alpha)q_{low} & \text{if sensitization level low} \\ (1+\beta)q_{norm} & \text{if sensitization level normal} \\ (1+\gamma)q_{high} & \text{if sensitization level high} \end{cases}$$

Where q_{low}, q_{norm} and q_{high} are the real success probabilities of the arc for a sensitization level, as extracted from Glorie, 2012.

Each method stands for a model with a different combination of these parameters, hence some are probably more fair and some which have more accurate success probabilities have higher outcomes. With these parameters we gradually approach the ratio of the deterministic case. For instance, by increasing the simulated success probability of high sensitization level patients. If we were to adjust the probabilities appropriately such that all three groups have the same success probability, we would expect the ratio between the amount of patients matched per group to be similar to the deterministic case, as illustrated by the example in Subsection VII-A. A logical success probability to take is the mean success probability of all arcs. We would then choose:

$$\alpha = \text{mean success probability}/0.94 - 1,$$
$$\beta = \text{mean success probability}/0.69 - 1,$$
$$\gamma = \text{mean success probability}/0.56 - 1$$

We call these values α_{fair}, β_{fair} and γ_{fair}. As this change in success probabilities to achieve complete fairness will most likely also decrease the amount of transplants, we also take values which approach this fairness. This is done by systematically adjusting the probabilities.

To systematically approach complete fairness, i.e. using α_{fair}, β_{fair} and γ_{fair}, we vary α, β and γ to be a proportion of those fair values. As all parameters influence one another, it would be very inefficient to adjust them all. Therefore, β only takes the proportions 0 and 1 of the fair value. α and γ nonetheless take proportions of the fair value between 0 and 1 with step size 0.2. Notice that if all parameters take proportion 0, this is the same as the failure-aware method we saw before without fairness in consideration.

We could also try to go even further, by trying to find success probabilities such that the chances for each sensitization level to be matched are equal. That would not only be tackling the unfairness created by failure-aware matching, but the overall unfairness. One should contemplate whether this really is 'fair', as one group is deliberately disadvantaged by advantaging another group. Moreover, it will decrease the number of eventual transplants. Because of the questionableness of the fairness in these adjustments and the even further decrease in transplants, we will keep ourselves to the first adjustments only, so striving for the ratio of the deterministic method.

C. Fairness Ranking

We have determined a way to rank the fairness of a solution. Say we have the ratios of Table I. The fairness number of this improved method is:

$$f = \frac{max(0,a-x)}{a} \cdot \frac{5}{3} + \frac{max(0,b-y)}{b} \cdot \frac{5}{3} + \frac{max(0,c-z)}{c} \cdot \frac{5}{3}$$

For each ratio, we look whether it has decreased or not for the new method. If it has decreased, than the maximum of 0 and the other number will be positive, otherwise it will be 0. This ensures that only if the ratio decreases will the number f increase. Therefore, if one ratio increases causing another ratio to decrease, this unfairness will not be counted twice. Furthermore, it is only unfair for the group whose ratio decreases. The maximum is divided by the ratio of the deterministic case, in order to scale the number. Therefore, a decrease in the ratio of a group is accounted for relatively. So a decrease of 5 percent in the low sensitization ratio equals more people than a decrease of 5 percent in the high sensitization ratio, however they have the same influence in the number f. Lastly, each fraction is multiplied by five-thirds to have numbers between 0 and 5. The difference between the fairness number f is in this case clearer for different methods than by taking a range between 0 and 1.

The higher f, the less fair the method. Theoretically, this can be a number between 0 and 5. Nevertheless, 5 would be an extremely high number, so most rankings

	Low Sensitization	Normal Sensitization	High Sensitization
Deterministic	a	b	c
Improved Method	x	y	z

Table I: Ratios for two methods

will be much closer to 0 than to 5 through failure-aware matching.

D. Results of the Fairness Model

The first results are the ratios between the probabilities of being matched per group. By having computed the fairness number, the results are ordered by this number. The ratios can be seen in Figure 19. The uppermost line of ratios represent the data of the deterministic case, therefore the fairness number is exactly equal to zero. The black vertical lines indicate the borders of these ratios. The lower you go in the graph, the higher the fairness number, hence the more the ratios deviate from the deterministic case. The low sensitization level patients never have a smaller ratio compared to the deterministic case, but in fact increase in ratio when the solution is less fair. The normal sensitization level patients do not deviate much from the deterministic ratio. In the first half of the less fair methods their ratio decreases somewhat. After that their ratio remains quite similar to the deterministic ratio. The high sensitization level patients however are disadvantaged a lot. Their ratio is never larger compared to the deterministic case, though only decreases a lot when the fairness decreases. In the first half of the less fair methods their ratio remains almost exactly the same, though in the second half a large decrease is established. Their largest decrease is by 38 percent in the least fair method.

The least fair combination is surprisingly enough not the case where all parameters are 0, this is yet the seventeenth unfair number from below, so 0.538. Gamma is low for all high unfair numbers, while the other parameters differ a lot more. We will also see this later that gamma has undoubtedly the most influence. Strangely there are no fairness numbers between 0.208 and 0.357. This is exactly the moment where the ratio of the high sensitization group significantly decreases. More different combinations of the parameters would likely result in more numbers in this range. The case where all parameters are equal to 1, thus all probabilities of failure are equal to the mean failure probability, is almost the most fair failure-aware case, i.e. the case with the lowest fairness number. It has fairness number 0.093, which is very close to the smallest fairness number 0.053. It is probably a measurement error that this case is not the most fair one, and more runs would have likely resulted in this case being the most fair one. For all cases with a rather low fairness number, β's proportion is still equal to 1, γ's proportion also is high, mostly 0.8 or 1, however α's proportion is mostly high as well but differentiates somewhat more between 0 and 1. While these numbers might be more fair, if the expected solution is much lower the case is not desirable. Therefore, Figure 20 shows the expected solution for each case, ordered by their fairness number again. None of the failure-aware cases cause a significant decrease in the expected amount of transplants. In the figure one can see a slight decrease in the expected solution. In the first few failure-aware cases there are two expected solutions which are probably by coincidence a bit lower than the rest. One should avoid such a situation where neither the fairness is high nor is the solution. Most however are around the value 46. Up to the fourth last method, the decrease is barely noticed. The fourth last method is only 2.6 percent less than the failure-aware method without fairness, i.e. the proportions are all zero. This equals 1.2 transplants. The last three failure-aware cases, so the ones which are the most fair, have approximately 3 less transplants than others. This is a decrease of about 6.5 percent compared to the least fair values. This is still 31.1 percent better than the deterministic case, compared to the value without fairness, which is 39.7 percent better here. One should contemplate whether to prefer the most fair case, or one a little less, which has a higher expected solution. We do see very clearly that all cases are still a lot better than the deterministic case.

Lastly, the influence of the different proportions separately can be observed in detail in Figures 21 and 22.

For proportion 0 of β_{fair} (Figure 21), α's proportion has almost no influence in the outcome. Only when γ's proportion is between 0.8 and 1 does α's proportion have an influence. When α's proportion then exceeds 0.8 one can see a steeper decline in the expected solution. γ's proportion on its own has the most influence in the expected solution when it is between 0.4 and 0.6. When the proportion of β_{fair} is 1 the figure changes. Overall, the results are lower than when β's proportion is 0. This is logically explained by the fact that the data is less accurate, as the normal sensitization group has as failure probability the mean and not its exact failure probability. In more depth, one can see that α's proportion has a bigger impact on the result now. Once the proportion is above 0.8, there is a steep decline in the amount of transplants occurring. Under 0.8, the proportion does not really matter for the result. Again γ's proportion always matters for the result, and seems to create the steepest decline in expected amount of transplants when it is between 0.4 and 0.6.

As one knows that for β's proportion 1 and γ's and α's proportion high the solution is more fair, it is very plausible that these numbers give a lower amount of expected transplants. More patients with higher failure probability are matched, thus there is a smaller chance of success. Also, the mean length of the cycles might be longer, as the mean success probability is quite high. As it is not yet obvious why these parameters influence the result in the way they do, we will look at what happens with the lengths of the cycles in the solution and the fairness number per parameter combination. When the mean length of the cycles increases, the solution decreases, as bigger sizes are more sensitive to failure. If the fairness number is lower, this means more patients with a higher failure-probability are matched, which also results in a lower outcome.

The mean length of the methods can be seen in Figures 23 and 24. For both proportion 0 and 1 of β_{fair} one can see that γ's proportion has a big influence. The higher the proportion, the longer the lengths of the cycle, with the steepest increases between 0.4 and 0.6. For α's proportion one does not see such an influence. There is even a minuscule decrease in length when α's proportion is big and γ's is not. Therefore, α's influence in Figure 22 cannot be clarified by the lengths of the cycles. γ's influence in the expected solution can be partly clarified by the increase in cycle lengths it causes.

Another factor could be the patients who are matched in the solution. In Figures 25 and 26 one can see the fairness number depending on the parameters. Again, one can see the evident influence of γ's proportion. The higher the proportion, the more fair the solution, i.e. the lower the fairness number. The most steep decline is also here between γ's proportion 0.4 and 0.6. α's proportion does not seem to have any influence when β's proportion is 0. However, when β's proportion is equal to 1, one does see a subtle influence of α's proportion. Especially when γ's proportion is also high. When α's proportion is above 0.8, there is a slight decrease in the fairness number, thus meaning the solution is more fair. While this decrease is relatively small, it does partially explain the decrease in the amount of transplants α causes in Figure 22. Nevertheless, it does not explain the influence fully as γ causes for a much bigger decrease in fairness number and a bigger increase in mean length of the cycles, while it does not cause for a substantially bigger decrease in the expected solution. Therefore, the reasoning of α's influence in the solution is not yet fully established.

E. Conclusion

In conclusion, one should consider by ethical reasoning which method to choose. The most-fair methods cause a decrease in expected solution of about 6.5 percent, while the fairness number is more than 6 times as low compared to the least-fair methods. The people which are disadvantaged the most through unfairness are the highly sensitized patients.

VIII. Results from the Internal Recourse Model

The earlier results were all based on the failure-aware method without recourse applied. Now, as a small analysis we will additionally look at the recourse model results. The weights are computed as described in Subsection V-C. With these results, the difference between the different methods can be observed, and one will see whether recourse really does result in better outcomes.

The results in this section are based on a data pool of 50 patient-donor pairs, and for 10 different instances. This is a lot less than before, which is because the algorithm had a relatively long runtime. The runtime can still be improved by implementing more specific cases in the algorithm or choosing the arcs even more wisely.

Figure 15 shows the ILP solutions and expected solutions for the Deterministic, Failure-Aware without recourse and Recourse model.

The 95 percent confidence intervals are quite somewhat higher than before, due to the low amount of different instances tested.

Similar to the failure-aware case, is the ILP solution the same as the expected solution for the recourse method. One can see that the recourse method achieves even better solutions. The expected solution of the recourse model is approximately 47.2 percent higher than the

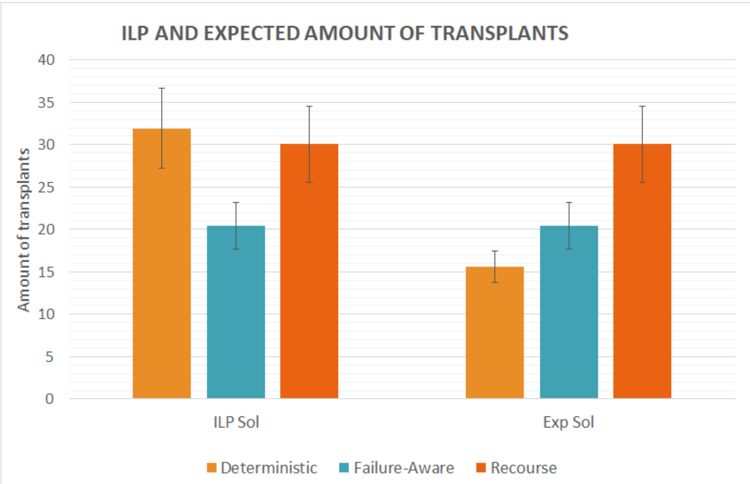

Figure 15: Solutions for the Deterministic, Failure-Aware and Recourse case

failure-aware model without recourse. Looking at each run separately, visualized in Figure 16, one can see again how the failure-aware case without recourse exceeds the deterministic case. Additionally, the recourse results are added. The orange bars represent the difference in outcome due to recourse compared to the elementary failure-aware case. The figure shows that the recourse result is substantially higher for each instance. This result is to be expected, as the recourse method allows for more real-life testing than the deterministic and failure-aware without recourse method. While these last two methods only test the arcs in the solution, the recourse method tests all internal arcs of the solution cycles. For a 4-cycle, this could mean testing 12 arcs instead of 4 arcs.

As could be observed in the previous figures there is a lot of evidence to conclude the recourse model is better. To strengthen this conclusion, again a paired t-test is executed on the outcomes of the recourse model and of the failure-aware model. The same hypotheses are applied, hence the null hypothesis is that the mean difference between the two values is zero, and the alternative hypothesis states that the difference in means is not zero. The p-value from the test is $3.675875 \cdot 10^{-6}$. This value is by far small enough to reject the null hypothesis and accept the alternative hypothesis. The mean difference between the two values is 9.641559 with the 95 percent confidence interval being [7.453419, 11.829698]. Thus, the recourse model results on average in 9.6 transplants more than the failure-aware model without recourse. In Figure 17 the difference between the two can once again be seen. The line between two dots represent the difference in outcome for the same instance. As not much instances were tested, there are not so many lines, nonetheless some slopes still seem to be a bit similar, meaning the increases are rather similar per run. For overall higher solutions we see that the line also becomes steeper, thus the increase caused by recourse compared to the simple failure-aware method becomes bigger. The standard deviation for the differences is approximately 3.1 transplants.

Lastly, we observe the cycle lengths used by the different methods. In Figure 18 one can see the mean amount of cycles used per length in a method for one instance. The deterministic and no recourse failure-aware method lengths have been observed earlier, and do not have unexpected results here. The recourse method is failure-aware, but does however have a lot of 4 cycles, a few 3 cycles and no 2 cycles. This is not surprising, as imagine there is a 2-cycle but this 2-cycle is also contained in a 4-cycle. Then the program would prefer to choose the 4-cycle, and if this cycle fails execute the 2-cycle in the case it succeeds. If the 2-cycle was chosen, no recourse could have been applied. Thus the method of recourse chooses longer cycles, as more possibilities for recourse are available in that case. One can deduce that for recourse, the higher the limit on the cycle size, the higher the expected outcome. However, as all internal arcs need to be tested, this also causes for a lot longer time duration of the process.

IX. Conclusion

The results show that failure-aware matching, in the no recourse setting, produces significantly higher outcomes than deterministic matching. For 100 patient-donor pairs with 50 different instances, the mean expected solution was 31.0 transplants for the deterministic case, while it was 44.2 transplants for the failure-aware case. This equals an increase of roughly 43 percent. The difference between both cases is on average 12.8 transplants, with most instances having a similar difference. Furthermore, we also see that the failure-aware method chose mostly 2-cycles for the solution, and no 4-cycles, while the deterministic method had a majority of 4-cycles in the solution. This confirms our predictions, as shorter cycles give higher expectations.

The failure-aware method did however cause a bigger inequality between the three sensitization groups. Lowly sensitized patients obtained a substantial higher probability in being matched, while for the highly sensitized patients this probability even decreased. The advantage of having a low PRA therefore becomes even more significant. Meanwhile, the high PRA's

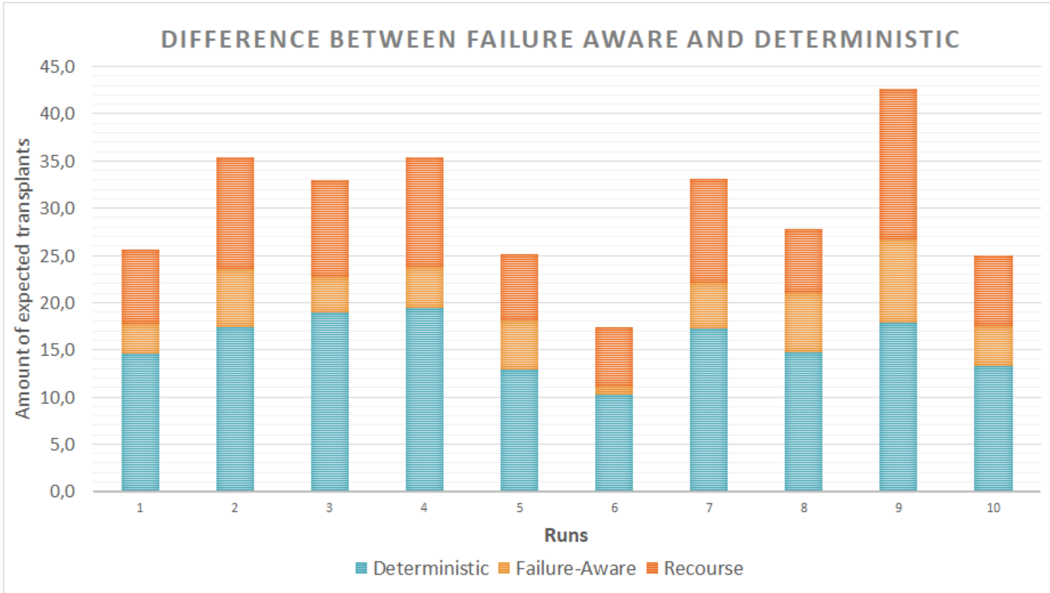

Figure 16: Difference between the expected outcome of the Deterministic, Failure-Aware and Recourse case per run

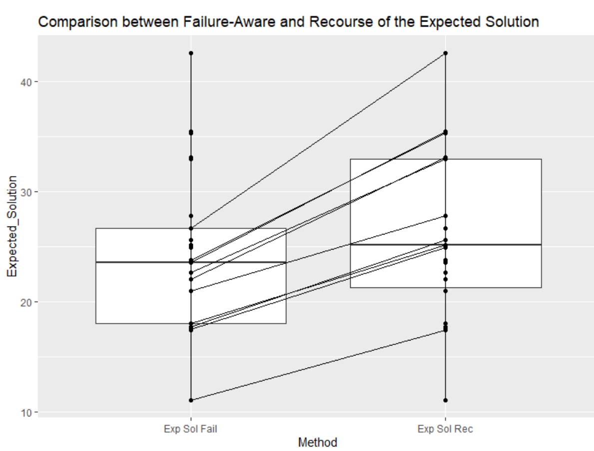

Figure 17: Difference between the expected solution of the Failure-Aware without recourse compared to the Recourse case

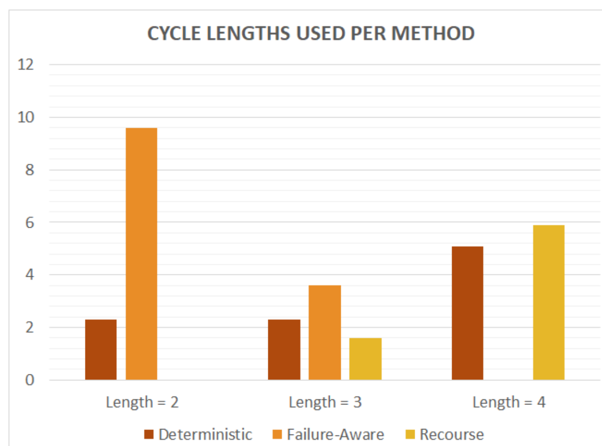

Figure 18: Distribution of the cycle lengths per method

are disadvantaged. By testing all the combinations of proportions of the fair parameters, the gradual influence of fairness could be seen. First, the normal sensitization group decreases in their ratio of the probability to be matched. After the first half of fair numbers, so for the last half of fair numbers, the normal sensitization ratio returns to its initial ratio, the deterministic one. Nevertheless, the high sensitization level patients decrease considerably in ratio, where the maximum reaches a decrease of 38 percent.

The decline in the amount of transplants reaches its maximum at about 6.5 percent. Most of the fair values lie very close together in expected solution, where the fourth fair method is only 2.6 percent worse than the failure-aware method without fairness. All fair cases still perform considerably better than the deterministic case, where the most fair case has a 31.1 percent higher expected solution.

Looking at how and why this expected solution differs for a method, we discovered that γ has the most influence. γ is the parameter for the high sensitization patients. For both proportions 0 and 1 of β_{fair}, the higher γ's proportion the lower the outcome. The reason for this is that for higher γ proportions, the mean length of the cycles increase and also the fairness number decreases. So longer cycles are formed due to the fact that the modeled success probabilities are higher, causing the program to consider the fact that the cycle will fail less. Also, more high sensitization patients are matched compared to other patients which causes there to be a lower real success probability. α's proportion does not have much influence when β's proportion is 0, however in combination of β's proportion being 1, high α proportions do have a lot of influence, it lowers the outcome substantially. A part of this can be explained by the lower fairness number caused by high α proportions when β's proportion is 1. However, this is very unlikely to be the entire cause, thus a part of the explanation is left unknown.

In the recourse setting, the results are significantly higher. The recourse model resulted in approximately 47.2 percent more transplants than the failure-aware model without recourse, which equals 9.6 transplants. Cycle lengths in the recourse solution are typically high, as the longer the cycle the more internal recourse opportunities.

X. Discussion

The main obstacle with stochastic optimization is the prerequisite of estimating the failure probability. These rates might be wrongly estimated, causing the model to be inaccurate. A slight deviation has no real harm, as we have seen with the fairness analysis that slightly other failure probabilities for the three groups do not influence the expected outcome that significantly. However, if these failure rates would be very imprecise, the solution might result in a lot less transplants than expected. If patients with high failure probabilities are estimated to have fairly low failure probabilities, and vice versa for patients with low failure probabilities, then the solution might contain mostly patients which have a low estimated failure probability but a high real failure probability. This will likely result in even worse outcomes than the deterministic case. Therefore, before applying stochastic optimization in real-life there should be enough thorough and adequate research executed on the failure rates of patients. Further research on how harmful deviations of the estimated failure rates are on the solution could be very interesting.

In the fairness analysis one can also see that the method which has the same failure probabilities for all three groups is still not completely fair. This could be caused by the possibility that high sensitization patients have a lower probability to be in 2-cycles than low sensitization patients have. Highly sensitized patients have a lot less arcs directed at them. Therefore, the probability that the patient-donor pairs they are connected to will form a 2-cycle is much lower than for lowly sensitized patients. Hence, not only equalizing the failure probabilities combats the unfairness of failure-aware matching entirely, as the cycles might also have an influence. We see however that the ratios do come very close to one another, so one needs not to worry that this is a great influence.

For the fairness analysis we also see in Figure 20 that some of the solutions are out of line with the expectation. A few outcomes are lower than the trend. The cause for this is probably inaccuracy, testing over more instances would likely resolve the problem. Unfortunately, due to runtime, we were not able to do so.

The unexpected influence of the proportion of α_{fair} on the expected solution cannot be fully explained. Further research on this aspect might be interesting, to be able to make a right and justifiable choice of parameters for one's method.

Furthermore, the algorithm 2 could be improved. In the algorithm, we have already slightly improved the Klimentova et al. 2016 version. In Klimentova et al. 2016, simply the first arc in the set of arcs is taken. This is a structural way of choosing arcs, in which an arc will always be drawn when the set is nonempty. However, we construct a more intelligent manner of choosing arcs, by first going through the arcs of the largest cycle in the internal cycles. If one of the arcs fail then the algorithm skips the rest of the arcs in the cycle and will check another internal cycle which can still succeed in the same manner. The order in which the cycles are chosen is by size, starting with the largest and ending with the smallest. However, one could explore this further, as the algorithm was still quite slow. For instance, the arcs could be chosen even more wisely. Instead of simply taking an arc of the corresponding cycle, an arc could be chosen which is in other large cycles as well. So the arc is taken which is in the first next cycle of *CyclesInternal* as well. If multiple arcs are in the same next cycle of *CyclesInternal*, we check for these arcs the next first cycle they are in as well. The arc chosen is hence the one which scraps the largest cycles from *CyclesInternal* if it fails, or causes the most arcs of the largest cycles to be determined to succeed if it succeeds, making the algorithm more efficient.

Moreover, there was unfortunately no time to do a fairness analysis involving recourse. Therefore, this is an interesting field to investigate further, as one cannot yet easily predict behaviour of the fairness models when recourse is applied. Expectation is, that it will be quite the same as with no recourse however then all results are a bit higher. Nevertheless, we are not sure of this, and other results might as well happen. For instance, a higher difference for less fair and more fair methods, as all internal cycles might fail due to the high failure probability.

Lastly, in this paper we chose not to add vertex failure nor chains to the research. Adding vertex failure would cause the results to approach real-life results more accurately. The difference in methods would nonetheless not change a lot. Vertex failure would be added likewise to arc failure. The expectation will then also include the failure probabilities of the vertices. In the recourse section the addition of vertex failure causes there to also be a set of vertices where the algorithm can draw from instead of only drawing from the set of arcs. Adding chains to the research would include adding non-directed donors to the model. The calculation of the expected value for chains is somewhat different, as when a failure occurs the pairs in the chain before this failure still receive a kidney. Recourse is also somewhat different for a chain. Thus, adding chains would affect the method of solving the model. As a continuation of this paper one could add these aspects of the kidney exchange program to make the model more realistic.

References

Abraham, D. J., Blum, A., & Sandholm, T. (2007). *Clearing Algorithms for Barter Exchange Markets: Enabling Nationwide Kidney Exchanges* (Tech. Rep.).

Biró, P., Haase-Kromwijk, B., Andersson, T., Ásgeirsson, E. I., Baltesová, T., Boletis, I., … Van Der Klundert, J. (2019). Building kidney exchange programmes in Europe - An overview of exchange practice and activities. *Transplantation*, *103*(7), 1514–1522. doi: 10.1097/TP.0000000000002432

Dickerson, J. P., Manlove, D. F., Plaut, B., Sandholm, T., & Trimble, J. (2016, 6). Position-Indexed Formulations for Kidney Exchange. *EC 2016 - Proceedings of the 2016 ACM Conference on Economics and Computation*, 25–42. Retrieved from `http://arxiv.org/abs/1606.01623`

Dickerson, J. P., Procaccia, A. D., & Sandholm, T. (2019, 4). Failure-Aware Kidney Exchange. *Management Science*, *65*(4), 1768–1791. doi: 10.1287/mnsc.2018.3026

Global Observatory on Donation and Transplantation. (2017). *Kidney Transplants.* Retrieved from `http://www.transplant-observatory.org/countkidney/`

Glorie, & K.M. (2012, 12). Estimating the probability of positive crossmatch after negative virtual crossmatch. *Econometric Institute Research Papers*. Retrieved from `https://ideas.repec.org/p/ems/eureir/38650.html`

Klimentova, X., Pedroso, J. P., & Viana, A. (2016, 9). Maximising expectation of the number of transplants in kidney exchange programmes. *Computers & Operations Research*, *73*, 1–11. Retrieved from `https://linkinghub.elsevier.com/retrieve/pii/S0305054816300533` doi: 10.1016/j.cor.2016.03.004

Maria, A., Carmelo, A. C. R., Taniar, D., Apduhan, B. O., & Hutchison, D. (2014). *Computational Science and Its Applications*. doi: 10.1007/978-3-319-09156-3

McElfresh, D. C., Bidkhori, H., & Dickerson, J. P. (2019, 7). Scalable Robust Kidney Exchange. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*(01), 1077–1084. Retrieved from `https://aaai.org/ojs/index.php/AAAI/article/view/3899` doi: 10.1609/aaai.v33i01.33011077

Nederlandse Transplantatie Stichting. (2020). *Organen: cijfers afgelopen maanden.* Retrieved from `https://www.transplantatiestichting.nl/publicaties-en-naslag/cijfers-over-donatie-en-transplantatie/organen-cijfers-afgelopen-maanden`

NHS. (2018). *ANNUAL REPORT ON LIVING DONOR KIDNEY TRANSPLANTATION* (Tech. Rep.).

Saidman, S. L., Roth, A. E., Sönmez, T., Ünver, M. U., & Delmonico, F. L. (2006). Increasing the opportunity of live kidney donation by matching for two- and three-way exchanges. *Transplantation*, *81*(5), 773–782. doi: 10.1097/01.tp.0000195775.77081.25

Working Group 2. (2020). National KEPs : models , optimisation methods and technology. *European Network for Collaboration on Kidney Exchange Programmes*(Handbook 1: A standardised framework for designing KEPs), 1–26.

Appendix

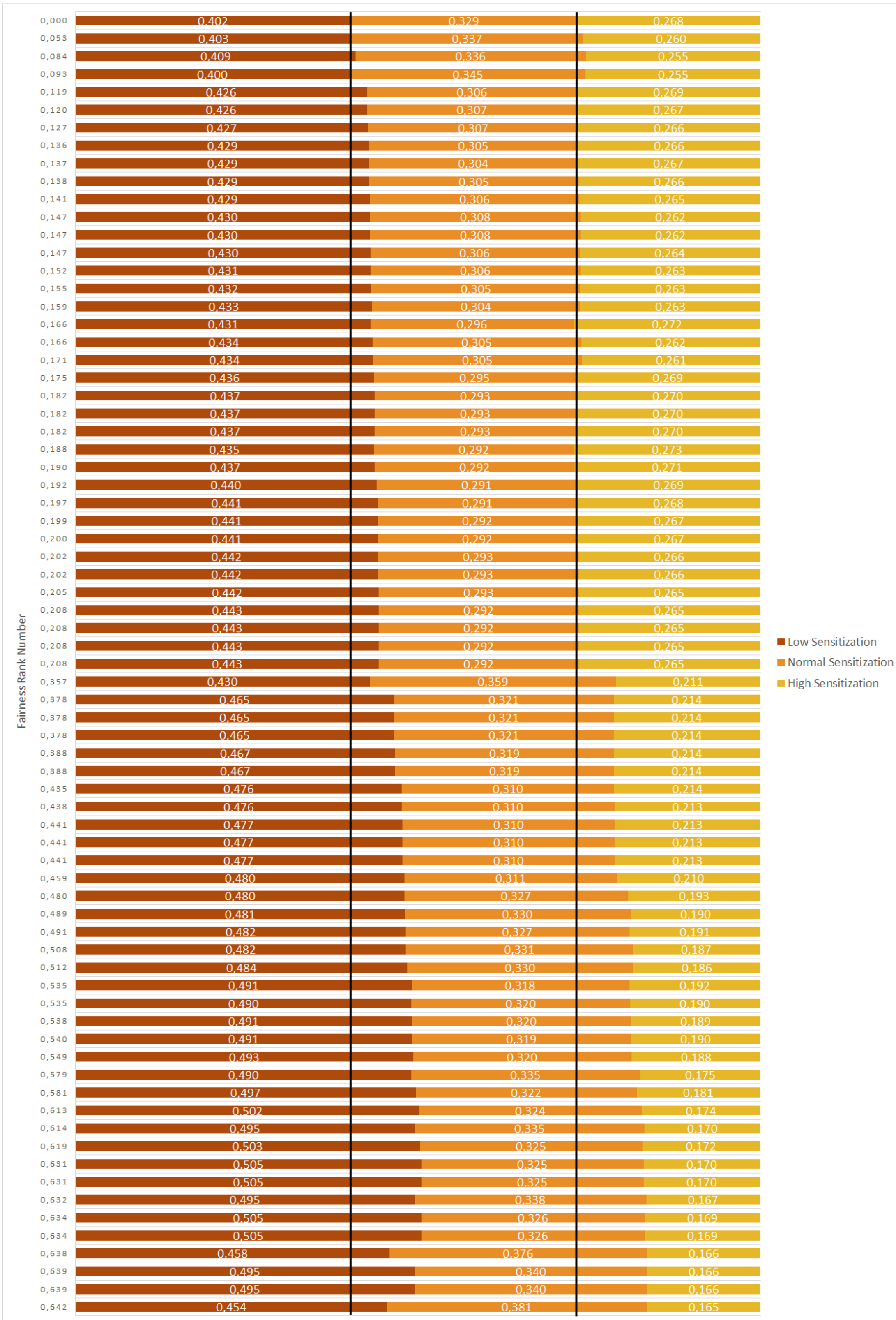

Figure 19: Ratios between probabilities of being matched ordered by ranking

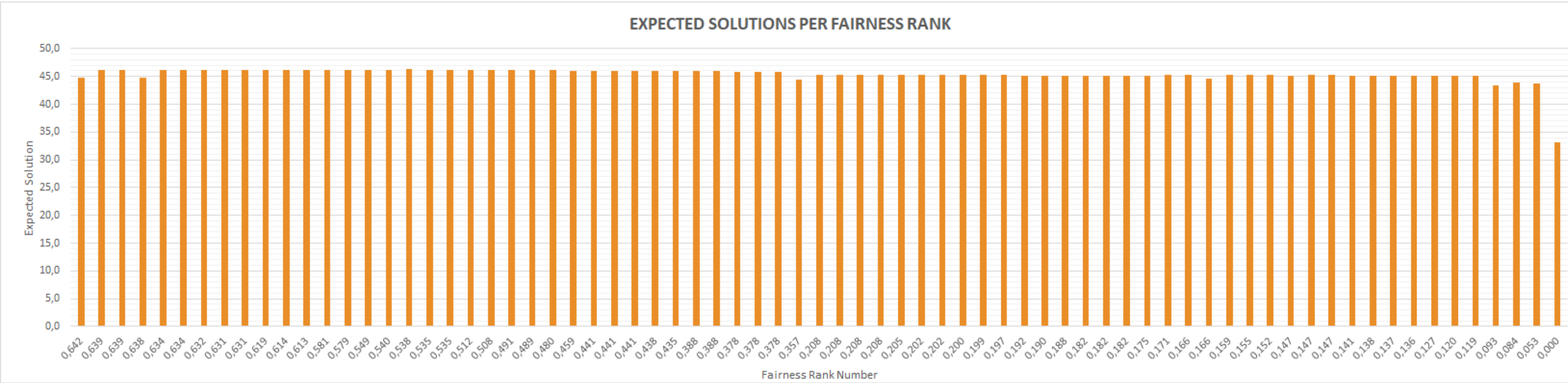

Figure 20: The expected amount of transplants resulting from a case

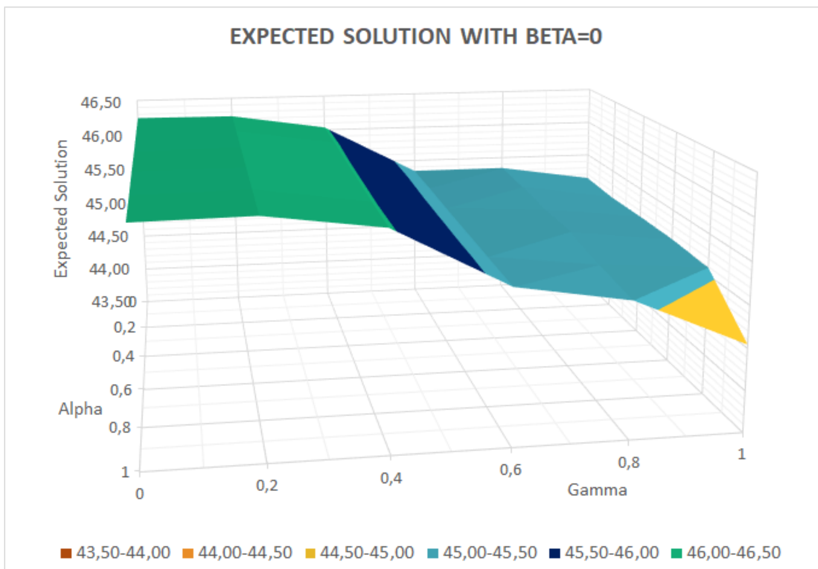

Figure 21: Expected solutions for different proportions of α_{fair} and γ_{fair}, with proportion 0 of β_{fair}

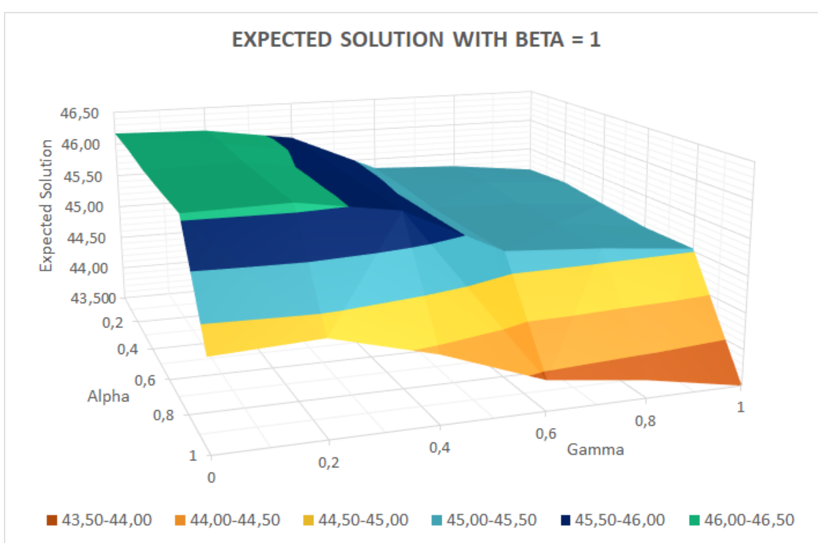

Figure 22: Expected solutions for different proportions of α_{fair} and γ_{fair}, with proportion 1 of β_{fair}

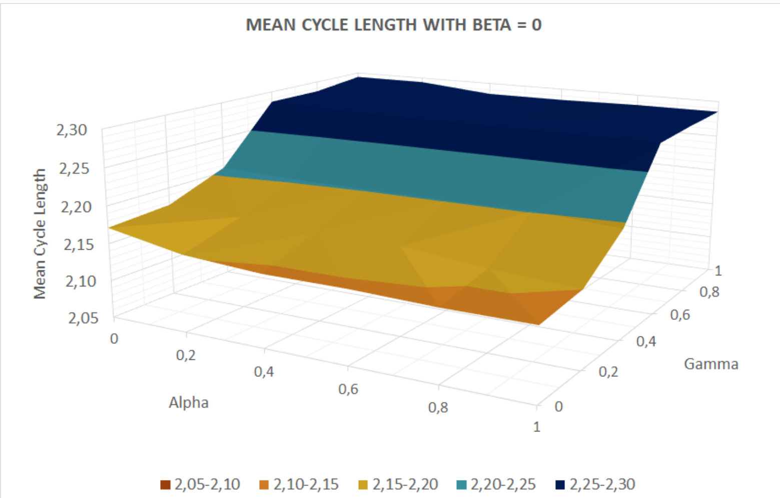

Figure 23: Mean Length of Solution Cycles for different proportions of α_{fair} and γ_{fair}, with proportion 0 of β_{fair}

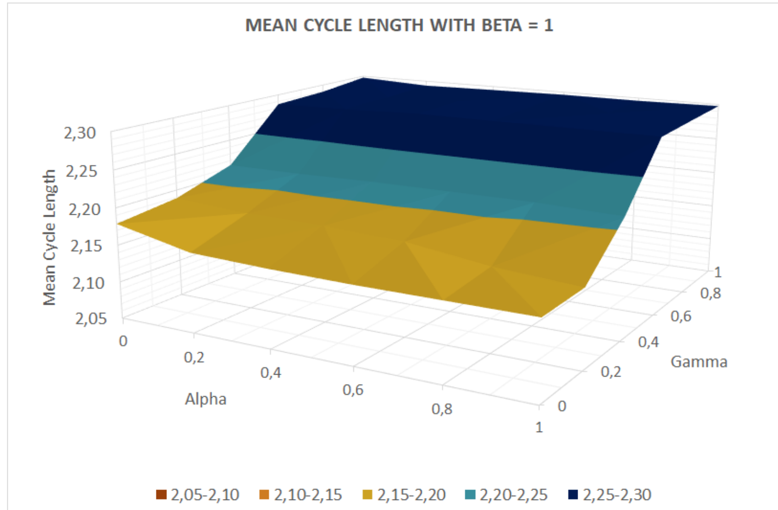

Figure 24: Mean Length of Solution Cycles for different proportions of α_{fair} and γ_{fair}, with proportion 1 of β_{fair}

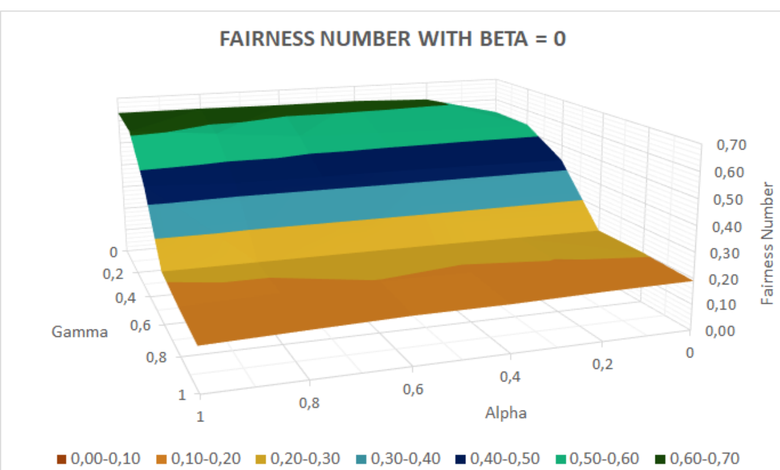

Figure 25: Fairness number of the solution for different proportions of α_{fair} and γ_{fair}, with proportion 0 of β_{fair}

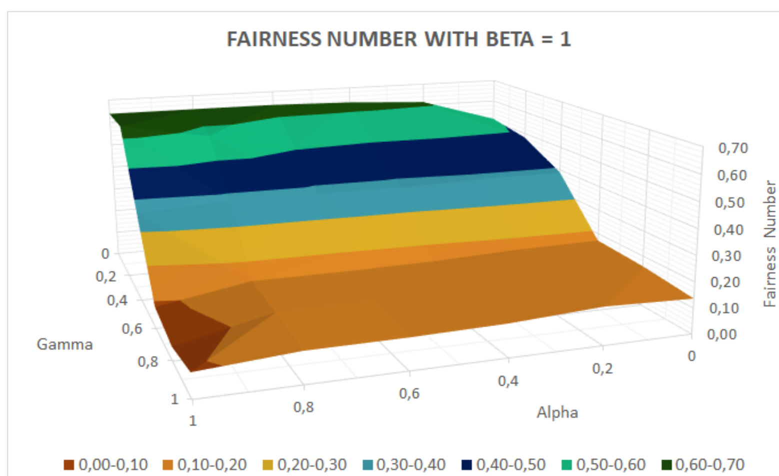

Figure 26: Fairness number of the solution for different proportions of α_{fair} and γ_{fair}, with proportion 1 of β_{fair}