Iterative branch-and-price for large multi-criteria kidney exchange

Kristiaan M. Glorie *[1] and Joris J. van de Klundert [2]

[1]Econometric Institute, Erasmus University Rotterdam
[2]Institute of Health Policy and Management, Erasmus University Rotterdam

June 5, 2013

Abstract

Living donor kidney transplantation is the preferred treatment for patients with end stage renal disease. Unfortunately, living donors are often incompatible with their specified recipient due to physiological reasons, such as incompatible blood types. Kidney exchange is an increasing modality that allows the exchange of kidneys between such incompatible donor-patient pairs. Typically, the aim is to find an allocation of donors to patients that is optimal with respect to multiple hierarchically ordered criteria. In this paper we show why existing approaches to the optimization of kidney exhange cannot deal effectively with multiple hierarchical criteria or with large, sparse, multi-center pools, which now begin to arise in practice. We then present a generic iterative branch-and-price algorithm which can deal with such multi-criteria exchanges and we show how the pricing problem can be solved in polynomial time for a general class of criteria. Our algorithm is effective even for large realistic donor-patient pools. Moreover, the algorithm accomodates inclusion of altruistic donors (who have no specified recipient) and individual rationality constraints for hospitals, as these are of increasing importance in clinical practice. Our approach and its effects are demonstrated using data from the Dutch national kidney exchange program, which is the oldest nationally coordinated program.

Key words: *Kidney Exchange, Multi-Criteria Optimization, Math Programming, Branch-and-price, Simulation*

1 Introduction

Living donor kidney transplantation is the preferred treatment for patients with end-stage renal disease. Unfortunately, in over 30 percent of the cases, patients

*Corresponding author: glorie@ese.eur.nl are incompatible with their specified living donor. Kidney exchange is a modality that identifies matches between such incompatible donor-patient pairs that allows them to proceed with transplantation: if the donor of a pair is compatible with the patient of a second pair, and the donor of the second pair is compatible with the patient of the first pair, the pairs can switch donors so that both patients can receive a transplant (e.g. Montgomery et al. (2006) and Klerk et al. (2011)). Exchange, in this way, need not be limited to two pairs but may involve cycles in which each donor gives a kidney to the patient of the next pair in the cycle. Also, unspecified donors - i.e. donors without a specified recipient - may initiate a chain of transplants in which the last donor is allocated to the deceased donor waitlist or is preserved for a future exchange. Such chains are increasingly common and important in clinical practice (e.g. Ashlagi et al. (2011), Glorie et al. (2012)).

Due to the large potential for increasing the number of transplants, many countries have begun developing kidney exchange programmes. Leading examples are the Netherlands, the US, the UK, Australia, and South Korea (Keizer et al. (2005), Manlove and O'Malley (2012), Park et al. (1999), Delmonico et al. (2004)). In 2004, the Netherlands succesfully launched the first national kidney exchange program in the world.

The design of kidney exchange programs raises the decision problem of assigning donors to recipients. In practice, this problem is typically considered in a *static* or offline context, in which exchanges are conducted at fixed time intervals and the assignment is optimized for the present population, as opposed to a *dynamic* or online context, in which exchanges are conducted continuously and the assignment is optimized with respect to the future evolution of the population. The difficulty of the decision problem arises from the requirement that all transplants in a cycle must be performed simultaneously, in order to prevent donors from reneging after their specified patient has received a transplant from another donor. This means that the length of a cycle is limited to the number of logistically feasible simultaneous transplants. Whenever this limit is finite and larger than 2, the kidney exchange problem is NP-complete (Abraham et al. 2007).

Abraham et al. (2007) present a mixed integer programming 'cycle' formulation for the single criterion kidney exchange problem with the objective of maximizing a weighted sum of cyclic transplants. They solve this formulation by a branch-and-price algorithm (see Barnhart et al. 1998), in which they identify positive price variables by depth-first search. Although alternative formulations for the kidney exchange problem have also been investigated, these have all been proven to be dominated by the cycle formulation (Constantino et al. 2013).

When exchanges are limited to cycles involving 3 pairs, Abraham et al. (2007) are able to solve large instances because the calculation of a tight upperbound (due to Roth et al. (2007)) allows them to quickly prove optimality and prevent the need to enumerate all positive price variables. We will show however, that as kidney exchange programs continue to evolve, this upperbound is no longer tight and - even in the case of a single decision criterion - the pricing problem becomes a major bottleneck. The reason for this is that the kidney exchange pools we begin to observe in practice are much sparser than assumed by Abraham et al. (2007) (see Ashlagi et al. (2011)), exchanges are not limited to 3 pairs but can involve - as altruistic donors allow for chains in which simultaneity is relaxed (see Roth et al. (2006)) - up to 6 or more pairs, and coordination between multiple transplant centers requires the implementation of individual rationality constraints (Ashlagi and Roth (2011) and Glorie et al. (2012)).

Moreover, in practice, maximizing the (weighted) sum of transplants is not the only relevant objective criterion (e.g. Klerk et al. (2011)). As the qualities of organs, the health of recipients, and the match between organs and recipients vary, so does the expected effect of transplantation. Also, certain recipients may be disadvantaged, e.g. because of poor blood type compatibility, which gives rise to equity or fairness considerations. Objective criteria related to these effects and considerations cannot always be modeled through a linear weight function (e.g. because no suitable interpretation can be given to the weights, or because a linear weight function does not suffice to capture the required structure) or it is not desirable to do so (e.g. because large differences in weights lead to numerical instabilities). Instead of a single weighted objective criterion, several existing kidney exchange programs use a hierarchically ordered set of criteria (e.g. Klerk et al. (2010), Manlove and O'Malley (2012), and Kim et al. (2007)). The Dutch national kidney exchange program, in particular, uses the following hierarchical set:

(i) Maximize the number of transplants;

(ii) Maximize the number of blood type identical transplants;

(iii) Match the patients in priority order based on 'match probability' (see Keizer et al. 2005);

(iv) Minimize the length of the longest cycle or chain;

(v) Maximize the spread over transplant centers per cycle and chain;

(vi) Match the patient with the longest waiting time.

The Dutch criteria are based on European agreements governed in the convention on human rights and biomedicine Council of Europe (2002), which determines that the allocation of organs should be both 'optimal' and 'fair'. For this reason the criteria include factors related to the probability of obtaining a transplant (criteria (ii) and (iii)) and waiting time (criterion (vi)). The exact aim of criterion (ii) is to help establish a fair allocation across patient blood types by ensuring that patients of disadvantaged blood types, such as blood type O, receive as many transplants as possible (donors of the same blood type will be reserved for them whenever this is viable). Criterion (iii) establishes such fairness in a broader sense by taking into account the total match probability (as defined in Keizer et al. (2005)). The priority order *within* criteria (iii) and (vi) is based on the traditional priority mechanisms for allocating deceased donor kidneys. Criteria (iv) and (v) are of a logistical nature. The hierarchy among the criteria implies that every criterion should be optimized subject to the best possible score on previous criteria. For example, the number of blood type identical transplants (criterion (ii)) should be maximized under the condition that the total number of transplants is maximum (criterion (i)).

Because of the evolution of kidney exchange pools and the ways in which exchange can take place, and because of the advent of large multi-center exchanges and the requirement of multi-criteria optimization, there is a need for new techniques for kidney exchange clearing. The work presented in this paper makes the following contributions:

1. We develop a generic iterative branch-and-price algorithm for clearing kidney exchanges with a hierarchically ordered set of objective criteria;
2. We propose a polynomial solution method for the pricing problems as they result for a general class of criteria (which includes all criteria of the Dutch exchange);
3. The presented approach encaptures long, possibly non-simultaneous, unspecified donor chains at practically feasible run times even in sparse exchange pools;
4. The approach allows for optimization for a set of transplantation centers, such as at a national level, while taking individual rationality constraints of the participating transplantation centers into account.

Our work builds on and extends recent work of Abraham et al. (2007), who introduced the cycle-formulation, and of Ashlagi and Roth (2011), who considered the individual rationality constraints in kidney exchange. Using simulations with actual Dutch kidney exchange data, we demonstrate the long term advantages of using our algorithm for multi-criteria kidney exchange clearing. To this end we implement the algorithm with the Dutch criteria listed above.

The paper is organized as follows. First, section 2 describes the multi-criteria kidney exchange problem mathematically. Section 3 details our iterative branch-and-price algorithm used to solve the multi-criteria kidney exchange problem. Section 4 discusses the simulator that allows us to analyze long term implications and section 5 presents the computational results. Finally, section 6 concludes the paper.

2 A kidney exchange model

In this section we formalize the concepts used in kidney exchanges and we mathematically define the problem under consideration.

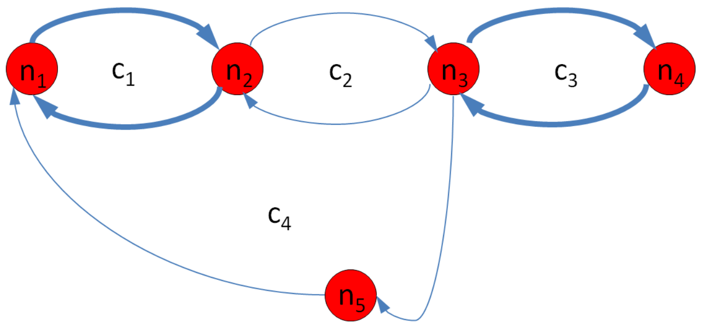

Figure 1: Kidney exchange example

2.1 Problem definition

Definition 1. *A kidney exchange pool N consists of two sets, i.e. $N = N_U \cup N_S$, where N_U refers to the set of all unspecified donors and N_S to the set of all incompatible specified donor-recipient pairs.*

Definition 2. *A kidney exchange graph $D = (N, A)$ has as its node set a kidney exchange pool N. There is an arc $a_{i,j} = \{n_i, n_j\} \in A$ from node $n_i \in N$ to node $n_j \in N_S$ if the donor corresponding to node n_i is compatible with the recipient corresponding to node n_j.*

Note that in any kidney exchange graph $D(N, A)$, nodes in N_U, which correspond to donors without recipients, have no incoming arcs. We define a transplant cycle and a transplant chain as follows

Definition 3. *In any given kidney exchange graph $D(N, A)$, a length-k cycle is an arc traversal $\langle n_1, \ldots, n_k \rangle$ such that $\{n_1, \ldots, n_k\} \subseteq N_S$ and such that $\{n_k, n_1\} \in A$ and, for every $1 \leq i < k$, $\{n_i, n_{i+1}\} \in A$.*

Definition 4. *A length l chain is an arc traversal $\langle n_0, \ldots, n_l \rangle$ such that $n_0 \in N_U$ and $\{n_1, \ldots, n_l\} \subseteq N_S$ and for every $0 \leq i < l$, $\{n_i, n_{i+1}\} \in A$.*

As, for all practical purposes, there exist exogenous limits on the number of transplants that can be performed simultaneously and in relation to eachother, we now firstly limit the sets of cycles and chains under consideration:

Definition 5. *For kidney exchange graph $D(N, A)$ and $K, L \in \mathbb{N}$,*

$$C(K, L) := \left\{ c \subseteq N : \begin{array}{l} c \text{ is a cycle in } D \text{ with length at most } K, \text{ or} \\ c \text{ is a chain in } D \text{ with length at most } L \end{array} \right\}$$

Note that, as chains can allow for the requirement of simultaneity to be relaxed, in general $L \geq K$.

Definition 6. *Let $D(N, A)$ be a kidney exchange graph, $K, L \in \mathbb{N}$, and $C(K, L)$ be defined as above. Then, any subset $M = \left\{c_1, c_2, \ldots, c_{|M|}\right\} \subseteq C(K, L)$, is called an exchange if $c_i \cap c_j = \emptyset$ for all, $1 \leq i, j \leq |M|$, $i \neq j$.*

Thus, an exchange is a collection of interdependent kidney transplantations which can be feasibly performed together. In the remainder, we assume a kidney exchange graph $D(N, A)$, and $K, L \in \mathbb{N}$ are given, and refer to $\mathcal{M}$ as the *exchange set*, i.e. the set of all exchanges M as defined above. Thus, an exchange set $\mathcal{M}$ always implicitly defines a kidney exchange graph $D(N, A)$, and $K, L \in \mathcal{N}$. Now that we have formally defined exchanges, and the exchange set, we proceed by considering the criteria by which exchanges $M \in \mathcal{M}$ are evaluated.

Definition 7. *For any given exchange set $\mathcal{M}$, a kidney exchange criterium is a function $f : \mathcal{M} \rightarrow \mathcal{R}$.*

We now arrive at the formal definition of the problem under consideration:

Definition 8. *For any given exchange set $\mathcal{M}$ and ordered set of kidney exchange criteria $\mathcal{I} = \{f_1, \ldots, f_{|\mathcal{I}|}\}$, a hierarchical multicriteria kidney exchange problem is to find an exchange $M^* \in \mathcal{M}$ such that, for each $i = 1, \ldots, |\mathcal{I}|$, $M^* \in \mathcal{M}_i$ where $\mathcal{M}_i$ is recursively defined as $\mathcal{M}_i := \{M \in \mathcal{M}_{i-1} : f_i(M) \geq f_i(M'), \forall M' \in \mathcal{M}_{i-1}\}$ with $\mathcal{M}_0 := \mathcal{M}$.*

Note that the set of criteria used in the Dutch kidney exchange program are an ordered set of kidney exchange criteria which fit the above definition, as would be the sole criterion of maximizing the number of transplants.

Figure 1 illustrates an example kidney exchange problem with 5 donor-recipient pairs, $n_1, \ldots, n_5$. The bound on the length of exchange cycles K is 4. The graph has 4 feasible cycles, $c_1 = \langle n_1, n_2 \rangle, c_2 = \langle n_2, n_3 \rangle, c_3 = \langle n_3, n_4 \rangle, c_4 = \langle n_1, n_2, n_3, n_5 \rangle$. There are two maximal exchanges given by $M_1 = \{c_1, c_3\}$ (highlighted) and $M_2 = \{c_4\}$. Although both exchanges have the same number of transplants, in the Dutch system exchange M_1 could be preferable over exchange M_2 by, for example, criteria (iv): the maximum cycle length is 2 instead of 4.

2.2 Integer programming formulations

In this paper we restrict our attention to kidney exchange problems that can be formulated as mixed integer linear programs. For example, for the single criterion of maximizing the number of transplants, the kidney exchange problem can be formulated using the so-called cycle formulation, which we describe below.

Pool size		Number of cycles and chains		
Pairs	Unspecified donors	Cycles ≤ 4	Chains ≤ 3	Chains ≤ 6
8	2	0	5.0e+1	5.4e+1
16	4	8	2.72e+2	1.06e+3
40	10	3.34e+2	6.31e+3	8.15e+5
80	20	1.32e+4	1.20e+5	3.29e+8
160	40	4.20e+5	3.18e+6	
400	100	2.19e+7	1.52e+8	

Table 1: Average number of cycles and chains over 5 random kidney exchange pools of the indicated size sampled from historical data of the Dutch national kidney exchange program

2.2.1 Cycle formulation

The cycle formulation due to Abraham et al. (2007) uses a binary decision variable x_c for each cycle and chain $c \in C(K, L)$ that is defined as:

$$x_c = \begin{cases} 1 & \text{if } c \in M^*, \\ 0 & \text{otherwise.} \end{cases}$$

Setting $x = \left[x_1, \ldots, x_{|C(K,L)|}\right]^T$, the integer program is given by:

P_0:

$$\max\ z_0(x) = \sum_{c \in C(K,L)} |c| \cdot x_c \tag{1}$$

$$\text{s.t.} \quad \sum_{c \in C(K,L): n \in c} x_c \leq 1 \qquad \forall n \in N \tag{2}$$

$$x_c \in \{0, 1\} \qquad \forall c \in C(K, L)$$

In P_0, the objective (1) is to select a collection of cycles and chains that maximizes the number of transplants. The constraints (2) ensure that no patient or donor is contained in more than one selected cycle or chain.

The number of variables in the cycle formulation can be very large (see Table 1 which shows the number of cycles and chains in pools based on actual data from the Dutch kidney exchange program), particulary because the number of chains grows rapidly with the number of nodes. In an exchange pool with 80 pairs and 20 unspecified donors there can be as many as 300 million chains up to length 6, thus the formulation requires at least that many variables. In contrast, Abraham et al. (2007) showed that, when dealing only with 2 and 3-cycles, this number of variables is only attained in pools of - roughly - 5,000 pairs or more (see Table 2 in Abraham et al. (2007)).

2.2.2 Generalized cycle formulation

It turns out that for each of the individual criteria (i)-(vi) mentioned in the introduction, as well as for many other practically relevant criteria, the single criterion kidney exchange problem can be modeled by the following generalized cycle formulation.

Consider a criterion $f_i \in \mathcal{I}$. As before, let $x = \left[x_1, \ldots, x_{|C(K,L)|}\right]^T$ denote the vector of decision variables that indicate whether a cycle/chain $c \in C(K, L)$ is selected. In addition, for $n_i, m_i \in \mathbb{N}$, let y_i denote a $n_i \times 1$ vector of auxilliary variables which are allowed to assume values in some subspace $F_i \subseteq \mathbb{R}^{n_i}$. Then, for $w_i \in \mathbb{R}^{|C(K,L)|}$, $v_i \in \mathbb{R}^{n_i}$, $A_i \in \mathbb{R}^{m_i \times |C(K,L)|}$, $B_i \in \mathbb{R}^{m_i \times n_i}$, and $b_i \in \mathbb{R}^{m_i}$, the generalized cycle formulation is given by the following integer program:

P_i:

$$\begin{aligned} \max\ z_i(x, y_i) = \quad & w_i^T x + v_i^T y_i && (3)\\ \text{s.t.} \quad & (2) && \\ & A_i x + B_i y_i \leq b_i && (4)\\ & x \in \{0,1\}^{|C(K,L)|} && \\ & y_i \in F_i && \end{aligned}$$

Here, the objective (3) is to maximize $z_i(x)$ with respect to f_i. As before, the constraints (2) ensure that no patient or donor is contained in more than one selected cycle or chain. The general constraints (4) allow for various relationships between the selected cycles x and the auxilliary variables y_i that are required to model f_i.

Illustration: Matching the patient with the longest wait time
To illustrate, we model here objective (vi), which is to match the patient with the longest wait time.

Let wait_n denote the wait time of the patient associated with node $n \in N_S$ and define

$$y_n := \begin{cases} 1 & \text{if patient } n \in N \text{ is the longest waiting patient that is matched,} \\ 0 & \text{otherwise.} \end{cases}$$

and

$$y'_n := \begin{cases} 1 & \text{if node } n \in N \text{ is matched,} \\ 0 & \text{otherwise.} \end{cases}$$

Then, letting $y_{(\text{vi})} = [y_1, \ldots, y_n, y'_1, \ldots, y'_n]^T$, the following integer program models objective (vi):

$$\begin{aligned}
\max\ z_{(\mathrm{vi})}(x, y_{(\mathrm{vi})}) = \sum_{n \in N} & \mathrm{wait}_n \cdot y_n && && (5)\\
\text{s.t.} \quad \sum_{c \in C(K,L): n \in c} & x_c = y'_n && \forall n \in N && (6)\\
& y'_n \leq y_n && \forall n \in N && (7)\\
& \sum_{n \in N} y'_n = 1 && && (8)\\
& x_c \in \{0,1\} && \forall c \in C(K,L) &&\\
& y_n \in \{0,1\} && \forall n \in N &&\\
& y'_n \in \{0,1\} && \forall n \in N &&
\end{aligned}$$

The objective (5) is to select a patient $n \in N$ with maximum waiting time. Constraints (6) have been derived from constraints (2) by including y'_n as a slack variable to indicate if node $n \in N$ is matched. Constraints (7) ensure that y_n can only be 1 if patient $n \in N$ is matched while constraint (8) ensures that only one patient can be the longest waiting patient.

Note, however, that the above integer program is not yet in the generalized cycle formulation. To translate the program to the generalized cycle formulation we should add the - in this case redundant - constraints (2), and relax the auxilliary variables y such that they are no longer integral (integral subsets are not subspaces of $\mathbb{R}$). We obtain the following mixed integer program (observe that the adjustments do not modify the solution space of the former integer program):

$$\begin{aligned}
\max\ z_{(\mathrm{vi})}(x, y_{(\mathrm{vi})}) = \sum_{n \in N} & \mathrm{wait}_n \cdot y_n && &&\\
\text{s.t.} \quad & (2), (6), (7), (8) && && (9)\\
& x_c \in \{0,1\} && \forall c \in C(K,L) &&\\
& y_n \in [0,1] && \forall n \in N &&\\
& y'_n \in [0,1] && \forall n \in N &&
\end{aligned}$$

2.2.3 Weighted criteria cycle formulation

Ostensibly, for a given set of criteria $\mathcal{I} = \{f_1, \ldots, f_{|\mathcal{I}|}\}$, the hierarchical multi-criteria kidney exchange problem can be formulated by combining the integer programs P_i into the following weighted criteria cycle formulation.

Let $\lambda_1, \ldots, \lambda_{|\mathcal{I}|} \in \mathbb{R}_+$. Then, using the notation introduced before, the weighted criteria cycle formulation is given by:

$$
\begin{aligned}
\max z = \quad & \sum_{i}^{|\mathcal{I}|} \lambda_i (w_i^T x + v_i^T y_i) && && (10)\\
\text{s.t.} \quad & (2) && \\
& A_i x + B_i y_i \leq b_i && i = 1, \ldots, |\mathcal{I}| \\
& x \in \{0,1\}^{|C(K,L)|} && \\
& y_i \in F_i && i = 1, \ldots, |\mathcal{I}|
\end{aligned}
$$

Here, the objective (10) is to maximize a weighted sum of the criteria $f_1, \ldots, f_{|\mathcal{I}|}$. By setting $\lambda_1, \ldots, \lambda_{|\mathcal{I}|}$ appropriately, all sorts of relationships between the criteria can be modeled. In particular, by setting the weights such that $\lambda_i \gg \lambda_{i+1}$ for $i = 1, \ldots, |\mathcal{I}| - 1$, the hierarchical relationship can be enforced.

One of the practical difficulties with the weighted criteria cycle formulation is that, for exchanges with six criteria as used in the Dutch national program, the required scaling of the weights leads to numerical instability which renders the program to be infeasible. Therefore, in the next subsection, we present a recursive formulation which models the criteria in the hierarchy without leading to numerical instability.

2.2.4 Recursive cycle formulation

In this subsection we present a new, recursive formulation modeling hierarchical criteria that does not suffer from such numerical instability: the *recursive cycle formulation.* The idea is not to encapture the hierarchical multi-criteria structure into a single integer pogram, but instead recursively define multiple programs $R_1, \ldots, R_{|\mathcal{I}|}$ which are linked together by 'objective propagation' constraints.

The first program in the recursion sequence is the generalized cycle formulation of criterion f_1. In case of the Dutch criteria, we have $R_1 := P_0$, where P_0 is the program we have defined before for the maximization of the number of transplants.

Then, denoting, in addition to the notation introduced above, the optimum value of R_i by z_i^*, the programs R_i, $i = 2, \ldots, |\mathcal{I}|$, are recursively defined as:

R_i:

$$
\begin{aligned}
\max z_i(x, y_i) = \quad & w_i^T x + v_i^T y_i && && (11)\\
\text{s.t.} \quad & (2) && \\
& A_j x + B_j y_j \leq b_j && j = 1, \ldots, i && (12)\\
& z_j(x, y_j) \geq z_j^* && j = 1, \ldots, i-1 && (13)\\
& x \in \{0,1\}^{|C(K,L)|} && \\
& y_j \in F_j && j = 1, \ldots, i
\end{aligned}
$$

As in the generalized cycle formulation, the objective (11) is to maximize the single criterion f_i. However, the constraints (12) now include all the relationships required for modeling criteria $f_1, \ldots, f_i$. Constraints (13) are the objective propagation constraints, which link the program R_i to the programs $R_1, \ldots, R_{i-1}$, by propagating their corresponding objective function values.

The recursive cycle formulation naturally fits the definition of the hierarchical multi-criteria kidney exchange problem. Indeed, the constraints (12) and (13) directly describe the sets $\mathcal{M}_i$, $i = 1, \ldots, |\mathcal{I}|$. The exchange corresponding to the solution of program $R_{|\mathcal{I}|}$ is the solution to the hierarchical multi-criteria kidney exchange problem.

2.2.5 Individual rationality constraints

Until now we have not yet considered the inclusion of individual rationality constraints required for multi-center coordination. In the integer programs we deal with, there are a class of constraints that are independent of the criteria under consideration.

Let H be the set of hospitals and transplant centers involved in the kidney exchange program. For each hospital $h \in H$, let $N_h \subseteq N$ be the subset of patients and donors associated with that hospital, and let A_h denote the compatibilities between them. When a hospital $h \in H$ does not participate in the national program, it can perform at most z_h^* transplants, where z_h^* is the solution to P_0 on the digraph $D_h = (N_h, A_h)$. In order to have no incentive to withhold patients or donors from the national program, the hospital should be able to perform at least z_h^* transplants when it fully participates in the national program. Hence, the individual rationality (or *participation*) constraints amount to:

$$\sum_{c \in C(K,L)} |c \cap N_h| \cdot x_c \geq z_h^* \qquad \forall h \in H \tag{14}$$

Note that, in this case, individual rationality refers to rationality of participating transplant centers, not of patients or donors or surgeons. Constraints (14) should be added to the integer programs modelling the various criteria under consideration.

For the recursive cycle formulation we will denote a program R_i appended with individual rationality constraints by IR_i.

3 Iterative solution approach

In this section we will develop an iterative branch-and-price algorithm for solving the hierarchical multi-criteria kidney exchange problem based on the recursive cycle formulation. The idea is to iteratively solve integer programs corresponding to the criteria. If a program is solved, its objective function value is propagated to the integer program corresponding to the next criterion by means

```
Step 0        Initialize $C(K,L)$ and $x = \left[x_1, \ldots, x_{|C(K,L)|}\right]^T$
FOR EACH      Hospital $h \in H$ DO
  |
  | Step 0.h  Solve $P_0$ on $D_h$:
  |           $z_h^* :=$  $\max_{x \in \{0,1\}^{|C(K,L)|}} f_0(x)$
  |                       s.t. (2)
  |
END FOR

FOR EACH      Criterion $f_i \in \mathcal{I}$ DO
  |
  |   Step i  Solve $IR_i$ on $D$:
  |           $z_i^* :=$  $\max_{x \in \{0,1\}^{|C(K,L)|}} f_i(x)$
  |                       s.t. (2), (14), $f_1(x) \geq z_1^*, \ldots, f_{i-1}(x) \geq z_{i-1}^*$
  |
END FOR

Output        $x^* :=$  argmax $_{x \in \{0,1\}^{|C(K,L)|}} f_i(x)$
                        s.t. (2), (14), $f_1(x) \geq z_1^*, \ldots, f_{i-1}(x) \geq z_{i-1}^*$
              $M^* := \{c \in C(K,L) : x_c^* = 1\}$
```

Table 2: Iterative algorithm for solving hierarchical multi-criteria kidney exchange

of an objective propagation constraint. Table 2 gives a schematic overview of this iterative approach, where IR_i and $f_i(x) := f_i(\{c \in C(K,L) : x_c = 1\})$ respectively denote the integer program corresponding to criterion f_i with individual rationality constraints, and the objective function value of the exchange corresponding to $x = \left[x_1, \ldots, x_{|C(K,L)|}\right]^T$ under criterion f_i.

3.1 Branch-and-price methodology

Because, the integer programming formulations described in Section 2 with one variable per cycle and chain, grow exponentially in the size of the exchange pool, the (recursive) integer programs $P_0, IR_1, \ldots, IR_{|\mathcal{I}|}$ included in the approach of Table 2 are solved using branch-and-price. The branch-and-price method starts with a limited subset $C \subseteq C(K,L)$ of cycles and chains and solves the LP relaxation of the integer program under consideration using the corresponding restricted variable set. Whenever linear programming duality conditions imply that adding not yet included variables may improve the solution value, corresponding cycles and chains are generated and added to C. This process repeats until strong duality conditions are satisfied. By doing this repeatedly for each node in a branch-and-bound tree, an optimal integral solution can be obtained.

Generating columns for any of the LP relaxations to $P_0, IR_1, \ldots, IR_{|\mathcal{I}|}$ corresponds to generating cycles and chains in the kidney exchange graph. Letting δ_n

denote the dual value of the constraint corresponding to node $n \in N$ in (2), for the LP relaxation of P_0, the reduced cost r_c of a cycle or chain $c \in C(K, L) \setminus C$ not yet contained in the problem are given by:

$$r_c = |c| - \sum_{n \in c} \delta_n$$

For the LP relaxation of IR_i, $i = 1, \ldots, |\mathcal{I}|$, we have the following generalized reduced cost r_c^i of a cycle or chain $c \in C(K, L) \setminus C$:

$$r_c^i = w_i[c] - \sum_{n \in c} \delta_n - \sum_{j=1}^{i} \sum_{k=1}^{m_i} A_j[k, c] \cdot \mu_{j,k} - \sum_{j=1}^{i-1} w_j[c] \cdot \nu_j - \sum_{h \in H} |c \cap N_h| \cdot \eta_h \quad (15)$$

where, as before, δ_n denotes the dual value of the constraint corresponding to node $n \in N$ in (2), $\mu_{j,k}$ denotes the dual value of the k-th constraint modelling criterion j in (12), ν_j denotes the dual value of the j-th objective propagation constraint in (13), and η_h denotes the dual value of the constraint corresponding to hospital $h \in H$ in (14).

In order to establish LP-optimality we search for cycles with positive reduced cost in the kidney exchange graph (see Section 3.3 for details on how this can be accomplished). If no such cycle can be found, the LP has been solved to optimality. If the LP solution is fractional, we branch, restricting one or more variables in the values they can assume, and then resolve the LP. At each node of the branching tree, the LP solution provides an upper bound on the restricted problem of that node. An integral lower bound can be obtained by solving the IP with the columns generated for the LP. If, at any node, the LP upper bound is no better than the best lower bound, its subtree can be pruned. If the IP lower bound matches the upper bound at the root node, the problem has been solved to optimality.

3.2 Branch-and-bound

3.2.1 Branching

An important and integral part of any branch-and-price procedure is the branching scheme. In the best branching scheme investigated in Abraham et al. (2007), branching is performed on the cycles (or, in our case, also chains) in the kidney exchange graph. Whenever the LP solution is fractional, the cycle whose corresponding variable has an LP value closest to 0.5 is selected and two branches are created, one in which the cycle's corresponding variable is set to 0, and one in which it is set to 1. Branches are then explored using depth-first search. As there are are up to $\sum_{i=2}^{K} |N|^i$ cycles of length K or less in D, the branching tree may have exponential depth. In our algorithm we branch on the arcs, of which there can be only up to N^2. We consider two branching schemes, based on the following definition:

Definition 9. *An arc $a \in A$ is fractional if*

$$x_a := \sum_{c \in C(K,L): a \in c} x_c$$

is fractional.

The existence of fractionally selected cycles need not immediately imply that a fractional arc exists. For instance, multiple fractional cycles might overlap, such that $x_a = 1$ for every arc $a \in A$. Fortunately, Theorem 1 establishes that this can never be true for all arcs whenever the LP solution is fractional.

Theorem 1. *There exists a fractional arc if and only if the LP solution is fractional.*

Proof. The first implication is trivial: if $a \in A$ is a fractional arc, then by definition of x_a, there must be at least one $c \in C(K,L) : a \in c$ for which x_c is fractional. To prove the other implication, suppose c_1 is a fractionally selected cycle containing arcs $a_1, a_2, \ldots, a_{|c_1|}$. If any arc $a \in c_1$ is not also covered by at least one other fractionally selected cycle, then $X_a = x_c$ and hence a is fractional. Therefore suppose there are one or more other fractional cycles which have at least one arc in common with cycle c_1. Now, let c_2 be such a fractional cycle containing, without loss of generality, arc $a_1 = \{n_1, n_2\}$ but not arc $a_2 = \{n_2, n_3\}$, and let $c_3, \ldots, c_m$ be all other fractional cycles containing arc a_1. There are two options: either $\sum_{i=1}^m x_c = 1$ or $\sum_{i=1}^m x_c < 1$. In the first case, $x_{a_1} = \sum_{i=1}^m x_c = 1$ so arc a_1, and hence node n_2, is totally covered, implying that no positively valued cycle $c \in C(K,L) \setminus \{c_1, c_3, \ldots, c_m\}$ can cover arc $a_2 = \{n_2, n_3\}$, and that therefore $x_{a_2} \in [x_{c_1}, 1 - x_{c_2}]$, making arc a_2 fractional. In the second case, $x_{a_1} = \sum_{i=1}^m x_c < 1$ and thus arc a_1 is fractional. This completes the proof. □

In our first branching scheme, we branch on groups of multiple arcs. If the LP solution is fractional, we select the node with the largest number of fractional out-arcs and then divide its out-arcs in two subsets, S_1 and S_2, and create a branch for each subset. In each branch, all the arcs of its corresponding subset are banned. The subsets $S1$ and $S2$ are determined by adding arcs to $S1$ in non-decreasing order of x_a value, until the sum of x_a values of arcs in $S1$ is at least 0.5. The remainder of the arcs are added to $S2$. Theorem 1 guarantuees us that we can always find a node with at least one fractional out-arc.

In the second branching scheme, we branch on only one arc at a time. If the LP solution is fractional, we select the arc with fractional value closest to 0.5. We then create two branches: one in which the arc is banned, and one in which it is enforced. Again, Theorem 1 guarantees that a fractional arc always exists, and, moreover, that when we have branched on all fractional arcs, we have an integer solution.

In order to enforce an arc $a \in A$ in the master problem, we need to add the following constraint:

$$\sum_{c \in C(K,L): a \in c} x_c = 1 \tag{16}$$

Adding constraint (16) to the master problem changes the reduced cost of a cycle or chain. In particular, if $A^* \subseteq A$ is the set of enforced arcs, the reduced cost r_c^i of a cycle or chain $c \in C(K,L) \setminus C$ in problem IR_i is now given by:

$$r_c^i = w_i[c] - \sum_{n \in c} \delta_n - \sum_{j=1}^{i} \sum_{k=1}^{m_i} A_j[k,c] \cdot \mu_{j,k} - \sum_{j=1}^{i-1} w_j[c] \cdot \nu_j - \sum_{h \in H} |c \cap N_h| \cdot \eta_h - \sum_{a \in A^*} \mathbf{1}_{a \in c} \xi_a \tag{17}$$

where, in addition to the previously introduced notation, ξ_a is the dual value of constraint (16) and $\mathbf{1}_{a \in c}$ is an indicator function which is 1 if $a \in c$ and 0 otherwise.

Note that banning an arc in the master problem is trivial, as that arc can simply be removed from the graph.

3.2.2 Bounding

In all cases, before branching, integral upper and lower bounds can be derived from the last iteration of the algorithm. For example, in Step 2, the maximum number of blood type identical transplants can not be higher than the total number of transplants determined in Step 1. Nor can it be lower than the number of blood type identical transplants in Step 1's solution. These derived bounds are used to prune the irrelevant parts of the branching tree as soon as they violate the bounds.

As in Step 1 there is no previous iteration, an upper bound can be derived by determining in polynomial time the maximum number of transplants when $L = K = \infty$. For graphs that are not too sparse, Roth et al. (2007) have shown that this upperbound is tight when the instance size is large. As in Abraham et al. (2007) such an upperbound can be determined by finding a maximum weight matching in a bipartite graph with donors on one side and patients on the other. Let us denote this bipartite graph as $G = (U, V, E)$, with U denoting the patients, V denoting the donors, and E denoting the edges. Donors are connected to their own patients with a zero-weight edge and to all other compatible patients with an edge of weight 1.

For each edge $e \in E$, let x_e be defined as:

$$x_e = \begin{cases} 1 & \text{if } e \text{ is selected,} \\ 0 & \text{otherwise.} \end{cases}$$

The maximum weight matching can then be found in polynomial time by solving the following LP:

$$
\begin{aligned}
\max \quad & \sum_{e \in E)} w_e \cdot x_e & \\
\text{s.t.} \quad & \sum_{e=\{u,v\} \in E} x_e = 1 & \forall u \in U \\
\text{s.t.} \quad & \sum_{e=\{u,v\} \in E} x_e = 1 & \forall v \in V \\
& x_e \in [0,1] & \forall e \in E
\end{aligned}
$$

During the branching process the initial bounds may be improved upon by the LP solutions (which provide an upper bound), or by a primal heuristic for constructing a feasible integer solution (which provides a lower bound). In all branching schemes, we use, as a primal heuristic, the solution to the IP with the columns generated for the LP. If, at any node of the branching tree, the LP upper bound is no better than the best lower bound, that node's subtree can be pruned. If, at any node, the IP lower bound matches the upper bound at the root node, the problem has been solved to optimality.

3.3 Pricing

In Abraham et al. (2007) the pricing problem is solved by traversing the kidney exchange graph D in search for a positive price cycle. In the worst case, this procedure enumerates all cycles in D and therefore (assuming $L \geq K$) is of order $\mathcal{O}(|N|^L)$, which is exponential in the size of the input. In this section we present a polynomial algorithm to solve the pricing problem in $\mathcal{O}(L\,|N|\,|A|)$.

The algorithm requires that the reduced cost of a cycle can be expressed as a linear function of arc weights. Therefore, we first formulate the following lemma on the reduced cost of a cycle or chain in the recursive cycle formulation.

Lemma 1. *If the objective coefficients $w_j[c]$ and the constraint coefficients $A_j[k,c]$, $j = 1,\dots,i$, $k = 1,\dots,m_j$ for each cycle or chain $c \in C(K,L)$ in problem IR_i can be described as linear functions of arc weights, then there exist weights $\pi^i_a \in \mathbb{R}$, for all arcs $a \in A$, such that, for every cycle and chain $c \in C(K,L)$,*

$$r^i_c = \sum_{a \in c} \pi^i_a \tag{18}$$

i.e. the reduced cost of c can also be described as a linear function of arc weights.

Proof. Let $w_i[c] = \sum_{a\in c} \alpha_i \omega_{i,a}$ and $A_j[k,c] = \sum_{a\in c} \beta_{i,j}\omega'_{j,k,a}$ for $j = 1,\dots,i$ and $k = 1,\dots,m_j$, then by (17),

$$
\begin{aligned}
r^i_c &= w_i[c] - \sum_{n\in c}\delta_n - \sum_{j=1}^{i}\sum_{k=1}^{m_i} A_j[k,c]\cdot\mu_{j,k} - \sum_{j=1}^{i-1} w_j[c]\cdot\nu_j - \sum_{h\in H}|c\cap N_h|\cdot\eta_h - \sum_{a\in A^*}\mathbf{1}_{a\in c}\xi_a\\
&= \sum_{a\in c}\alpha_i\omega_{i,a} - \sum_{n\in c}\delta_n - \sum_{j=1}^{i}\sum_{k=1}^{m_i}\sum_{a\in c}\beta_{i,j}\omega'_{j,k,a}\cdot\mu_{j,k} - \sum_{j=1}^{i-1}\sum_{a\in c}\alpha_j\omega_{j,a}\cdot\nu_j\\
&\quad - \sum_{h\in H}|c\cap N_h|\cdot\eta_h - \sum_{a\in A^*}\mathbf{1}_{a\in c}\xi_a\\
&= \sum_{a=\{n,n'\}\in c}\left(\alpha_i\omega_{i,a} - \delta_{n'} - \sum_{j=1}^{i}\sum_{k=1}^{m_i}(\beta_{i,j}\omega'_{j,k,a})\cdot\mu_{j,k} - \sum_{j=1}^{i-1}(\alpha_j\omega_{j,a})\cdot\nu_j - \eta_{h(n')} - \mathbf{1}_{a\in A^*}\xi_a\right)\\
&= \sum_{a=\{n,n'\}\in c}\pi^i_a
\end{aligned}
$$

where

$$
\pi^i_a = \alpha_i\omega_{i,a} - \delta_{n'} - \sum_{j=1}^{i}\sum_{k=1}^{m_i}(\beta_{i,j}\omega'_{j,k,a})\cdot\mu_{j,k} - \sum_{j=1}^{i-1}(\alpha_j\omega_{j,a})\cdot\nu_j - \eta_{h(n')} - \mathbf{1}_{a\in A^*}\xi_a \tag{19}
$$

with δ_n the dual value of the constraint (2) for node n, $\mu_{j,k}$ the dual value of the k-th constraint modelling criterion j in (12), ν_j the dual value of the j-th objective propagation constraint in (13), $\eta_{h(n)}$ the dual value of constraint (14) for the hospital $h \in H$ associated with node n, and ξ_a the dual value of constraint (16). □

The linear relationship between the objective and constraint coefficients and the arcs in D holds for most criteria used in practice. In particular, all of the Dutch criteria have this property. Also, the constraints required for branching on arcs have this property. Note, however, that the constraints required to branch on cycles (as used by Abraham et al. (2007)) do not satisfy this relationship, as they require constraints to enforce the inclusion of a single cycle.

Now, let us define a reversion operator as follows:

Definition 10. *For any directed cycle or chain $c = \langle n_1, n_2, \dots, n_{|c|}\rangle$, the directed cycle (resp. chain)*

$$c^{-1} := \langle n_{|c|}, n_{|c|-1}, \dots, n_1\rangle$$

is the reverse of c.

The pricing problems can now be solved in polynomial time through the algorithm given in Table 3. The algorithm first constructs the arc set $\widetilde{A} \subseteq A$ of arcs that are not banned and then determines for each starting node $n \in N$ a shortest path up to length K or L in $\widetilde{D} = (N, \widetilde{A})$ (depending on whether node n corresponds to an unspecified donor or not) using an adapted version of the Bellman-Ford method (Bellman 1958)(Ford 1956). For each node $n \in N$ and $k = 0, 1, \ldots, K-1$ ($L-1$ for chains) the algorithm calculates functions $f_k^n : N \to \mathbb{R} \cup \{\infty\}$ and $g_k^n : N \to N$ that respectively provide the weight of the shortest path between n and any other node $n' \in N$ using at most k arcs, and the predecessor of node $n' \in N$ on such a shortest $n - n'$ path.

The algorithm consists of four main steps. Before executing the main steps, Step 0 transforms the arc specific weights obtained from Lemma (1) such that the pricing problem becomes a minimization problem. Then, for each node $n \in N$, Step 1 initializes the functions f_k^n and g_k^n, Step 2 calculates the function values of f_k^n and g_k^n in case of a cycle (i.e. $n \in N_S$), and Step 3 calculates the function values in case of a chain (i.e. $n \in N_U$). The final step, Step 4, checks whether there are cycles or chains with positive reduced cost and, if there are, reversely constructs them from the function values of g_k^n.

As stated in Theorem 2 below, the algorithm is exact, i.e. it always finds a positive price cycle or chain if one exists. In fact, for each starting node it finds the maximum weight cycle of length at most K (or chain of length at most L). However, it might be the case that a cycle or chain returned by the algorithm contains a subcycle (and hence is not feasible for the master problem). In the case of such a compound cycle or chain, Theorem 2 guarantuees us that the subcycle will always have a positive price. We can choose to abort the algorithm as soon as a positive price cycle or chain is found, or it can be run to completion, possibly resulting in multiple positive price cycles and chains being identified (NB. If run to completion, the algorithm will output each cycle c^* up to $|c^*|$ times, therefore it may be desirable to filter the generated cycles for duplicates).

Before providing the theorem, we first introduce the following definition:

Definition 11. *For any directed cycle c composed of simple cycles $\sigma_1, \ldots, \sigma_m$ in $D = (N, A)$, and arc weights $\pi_a^i\ \forall\ a \in A$, the maximum simple cycle $S(c)$ is the cycle given by*

$$S(c) = \mathit{argmax}_{\sigma \in \{\sigma_1, \ldots, \sigma_m\}} \left\{ \sum_{a \in \sigma} \pi_a^i \right\}$$

Theorem 2. *$C^* \neq \emptyset$, and, for all $c^* \in C^*$, $S(c^*) \in C(K, L)$ and $r_{S(c^*)}^i > 0$, if and only if $\exists c \in C(K, L) : r_c^i > 0$.*

Proof. Analogous to the Bellman-Ford method, we have, for each $n, n' \in N_S$, $k = 0, \ldots, K$, that

$$\begin{aligned} f_k^n(n') &= \min \left\{ \sum_{a \in P} w'_a : P \text{ is an } n - n' \text{ walk traversing at most } k \text{ arcs} \right\} \\ &= \max \left\{ \sum_{a \in P} w_a : P \text{ is an } n - n' \text{ walk traversing at most } k \text{ arcs} \right\} \end{aligned}$$

```
Step 0      Set $w'_a := -\pi^i_a \ \forall\ a \in \widetilde{A}$ as in (19), $C^* = \emptyset$
FOR EACH    Node $n \in N$ DO
    Step 1  Set $f^n_0(n) := 0$ and, $\forall\ n' \in N \backslash \{n\}$, $f^n_0(n') := \infty$ and $g^n_0(n') := \emptyset$
    Step 2  IF $n \in N_S$ THEN
                set, for $k = 0, \dots, K-2$, and for all $n' \in N$,
                $\hat{a} = \{n'', n'\} := \operatorname{argmin}_{a=\{u,n'\}\in\widetilde{A}} \{f^n_k(u) + w'_a\}$,
                $f^n_{k+1}(n') := \min\{f^n_k(n'), f^n_k(n'') + w'_{\hat{a}}\}$,
                $g^n_{k+1}(n') := \begin{cases} n'' & \text{if } f^n_k(n'') + w'_{\hat{a}} < f^n_k(n'), \\ g^n_k(n') & \text{otherwise.} \end{cases}$
    Step 3  ELSE IF $n \in N_U$ THEN
                set, for $k = 0, \dots, L-2$, and for all $n' \in N$,
                $\hat{a} = \{n'', n'\} := \operatorname{argmin}_{a=\{u,n'\}\in\widetilde{A}} \{f^n_k(u) + w'_a\}$,
                $f^n_{k+1}(n') := \min\{f^n_k(n'), f^n_k(n'') + w'_{\hat{a}}\}$,
                $g^n_{k+1}(n') := \begin{cases} n'' & \text{if } f^n_k(n'') + w'_{\hat{a}} < f^n_k(n'), \\ g^n_k(n') & \text{otherwise.} \end{cases}$
END FOR
Step 4      For $n, n' \in N_S$, if $\{n', n\} \in \widetilde{A}$ and $f^n_{K-1}(n') + w'_{\{n',n\}} < 0$,
            $C^* \to C^* \cup \left\{n', g^n_{K-1}(n'), g^n_{K-2}(g^n_{K-1}(n')), \dots, n\right\}^{-1}$,
            and, for $n \in N_U, n' \in N_S$, if $f^n_L(n') < 0$,
            $C^* \to C^* \cup \left\{n', g^n_{L-1}(n'), g^n_{L-2}(g^n_{L-1}(n')), \dots, n\right\}^{-1}$
```

Table 3: Polynomial pricing algorithm

Then, obviously,

$$\begin{aligned} c^*(n) &:= \{n', g^n(n'), g^n(g^n(n')), \dots, n\}^{-1} \\ &= \operatorname{argmax} \left\{ \sum_{a \in P} \pi_a^i : P \text{ is an } n-n \text{ walk traversing at most } k \text{ arcs} \right\} \end{aligned} \tag{20}$$

is a, possibly compound, maximum weight cycle with length at most K. Let $\sigma_1(n), \dots, \sigma_m(n)$ be the simple cycles composing $c^*(n)$ (if $c^*(n)$ itself is a simple cycle, $m = 1$ and $\sigma_1(n) = c^*(n)$). By definition, $S(c^*(n)) \in \{\sigma_1, \dots, \sigma_m\} \subseteq C(K, L)$. Therefore, it remains to prove that $\exists n \in N_S : c^*(n) \in C^*$ and that, for all $n \in N_S : c^*(n) \in C^*$, $\sum_{a \in S(c^*(n))} \pi_a^i > 0$.

To prove the first part, let $c \in C(K, L)$ be a cycle with $\sum_{a \in c} \pi_a^i > 0$ and let $n \in c$. By (20) we then have that $\sum_{a \in c^*(n)} \pi_a^i \geq \sum_{a \in c} \pi_a^i > 0$, and, therefore

$$f^n_{K-1}(n') + w_{\{n',n\}} = \sum_{a \in c^*(n)} w'_a = -\sum_{a \in c^*(n)} \pi_a^i < 0$$

which implies that $c^*(n) \in C^*$ as desired.

To prove the second part, let $n \in N_S : c^*(n) \in C^*$. Then

$$\sum_{a \in c^*(n)} \pi_a^i = \sum_{a \in \sigma_1(n)} \pi_a^i + \dots + \sum_{a \in \sigma_m(n)} \pi_a^i > 0.$$

Because of this, $\exists \sigma \in \{\sigma_1(n), \dots, \sigma_m(n)\} : \sum_{a \in \sigma} \pi_a^i > 0$, and, by definition 11, $\sum_{a \in S(c^*(n))} \pi_a^i > 0$ as desired. The proof for chains is analogous. □

Corollary 1. *Given a kidney exchange graph $D = (N, A)$ and arc weights π_a^i $\forall\ a \in A$, a positive weight cycle up to length K or chain up to length L, if one exists, can be found in time $\mathcal{O}(\max\{K, L\}\,|N|\,|A|)$.*

Proof. Directly from the description of the algorithm in table 3. □

4 Simulators

We test our algorithm using two realistic simulators. The first is a kidney exchange simulator based on historical data from the Dutch national kidney exchange program. This simulator is described in detail in (Glorie et al. 2012). We use this simulator to both generate static kidney exchange pools (individual pools sampled from the available patient-donor population) as well as dynamic sequences of pools and exchanges (pools that dynamically evolve by simulating arrivals sampled from the patient-donor population and by simulating removals due to exchanges and, for example, patient illness). In order to also generate instances with different characteristics and larger size, we use a second simulator that is a slightly modified version of the simulator described in (Saidman et al. 2006) (and used in (Abraham et al. 2007)), which is the most commonly used generator for kidney exchange pools. It is based on US population data. We make some modifications to the simulator in order to capture new insights into the composition of exchange pools in practice described in (Ashlagi and Roth 2012). We use this simulator to generate an alternative set of static kidney exchange pools. In this section we will briefly explain the main aspects of the data and simulation procedures.

4.1 Static simulation with actual Dutch data

The data for our first simulator is obtained from the Dutch Transplant Foundation (NTS) and originates from the empirical registry of the Dutch national kidney exchange program. It includes 438 incompatible patient-donor pairs who participated in Dutch kidney exchanges between October 2003 and January 2011. In addition it contains 109 unspecified donors who were screened at one of the seven Dutch transplant centers during that period. Each patient and donor has a blood type as well as a registration center. Donors also have a record of their so-called human leukocyte antigen (HLA) types, while patients have a record of the HLA types that are medically unacceptable to them. A patient is marked as incompatible with a donor whenever the donor's blood type contains a protein that is not contained in the patient's blood type, or whenever the donor has a HLA type that is unacceptable to the patient, otherwise the patient and donor are compatible. A static kidney exchange pool is generated at random from the data using sampling with replacement.

4.2 Dynamic simulation with actual Dutch data

We use the static simulator described above to perform dynamic kidney exchange simulations as described in (Glorie et al. 2012). The dynamic simulation procedure consists of repeated Monte Carlo simulations. Each simulation spans the period between 1 October 2003 and 23 December 2010 and involves a population of size 547 generated from the empirical data using sampling with replacement. The arrivals of patient-donor pairs and unspecified donors are determined by assigning each pair and each unspecified donor in the sampled population a random date in the simulation period. Arrival dates are drawn uniformly, corresponding to a Poisson arrival process. Matching rounds are conducted every three months, starting from 1 January 2004. In each matching round, the optimization algorithm described in Section 3 implemented with the Dutch hierarchical criteria identifies a matching. There are a total of 29 matching rounds during the simulation period. Proposed matches may fail with a probability depending on the patient and donor characteristics, in which case the optimization algorithm is rerun with the new information. This process is repeated until a feasible matching is found. Patients and donors may leave the pool over time due to simulated attrition and reneging.

4.3 Static simulation with US population data

We also perform simulations with US population data using the simulator described in (Saidman et al. 2006). The simulation is based on data from the United Network for Organ Sharing (UNOS) in the US. The simulator generates patients with a random blood type, sex, and probability of being crossmatch incompatible with a randomly chosen donor. Each patient is assigned a potential donor with a random blood type and relation to the patient. If the patient and the potential donor are incompatible, they are added to the kidney exchange pool. Blood types and probabilities of crossmatch failure are then used to determine the compatibilities in the pool.

As Ashlagi and Roth (2012) recently found that the percentage of highly sensitized patients (i.e. patients with a high probability of crossmatch incompatibility with a randomly chosen donor) in practice is significantly higher than assumed in (Saidman et al. 2006), we modify the final pool using a distribution for patient sensitization that characterizes the empirical distribution described in (Ashlagi et al. 2013): half of the patients is assigned a high sensitization level (a probability of crossmatch incompatibility of 97.5 %) and the other half is assigned a low sensitization level (a probability of crossmatch incompatibility of 2.5 %). Final compatibilities are then determined using the modified sensitization levels. This means that the pools are much sparser than the pools generated by the original simulator due to Saidman et al. (2006).

Additionally, because we want to model multi-center exchanges, we randomly assign the patients and donors to a transplant center. To ensure a realistic distribution of center sizes, we take the empirical distribution of center size obtained from our first simulator.

5 Computational results

Our experiments were performed on a Windows 7 64 bit computer with an 3 GHz AMD Athlon II X2 processor and 4 GB of RAM. The iterative branch-and-price algorithm has been implemented in C#.NET and LP's and ILP's are solved using CPLEX 12.5.

Table 4 displays the run time performance of our algorithm for solving the usual primary objective (maximizing the number of transplants) on instances constructed by the simulator with Dutch clinical data described in Section 4. The performance of the different pricing and branching strategies described in section 3 is compared on instances of various sizes. The cycle length limit K is set to either 3 (short cycles) or 4 (long cycles) and the chain length limit L is set to either 3 (short chains) or 6 (long chains).

In our comparisons we include the depth-first pricing algorithm with cycle branching used in Abraham et al. (2007). In this algorithm, the kidney exchange graph is traversed for positive price cycles by exploring nodes in non-decreasing dual value order. Intermittently, the search path is pruned based on the fact that new nodes will have dual value as least as large as the current node.

In all instances containing more than 400,000 cycles and chains the master problem is seeded with a starting collection of 10,000 random cycles and chains (generated by random walks from a randomly chosen node in the kidney exchange graph until a feasible cycle or chain is found). The collection of cycles and chains is managed such that whenever the problem contains more than 400,000 cycles and chains, the cycles and chains with the lowest reduced cost are deleted (excepting those that are branched on or have a non-zero LP value).

Per pricing iteration up to 100 new cycles and chains are added (except in the depth-first pricing algorithm, where we adhered to the setting of 1 new cycle or chain per iteration, as advised in Abraham et al. (2007) and which, after tuning, we found to work best for this pricing alorithm).

The first column in Table 4 indicates the pool size. The second column contains the total run time in seconds. The third and fourth column respectively contain the time spent on solving LP's and IP's for the master problem. When branching is applied, the fifth column reports the number of processed nodes in the branch-and-bound tree over the total number of nodes in the tree; the sixth column reports the total time required for solving pricing problems.

As can be seen from the table, our algorithm is able to find optimal solutions in instances with 500 nodes – which can contain up to $100 \cdot 400^6 \approx 4.10\text{e}{+}17$ chains of length 6 – within two minutes. In almost all instances the polynomial pricing algorithm performs better than the depth-first pricing algorithm. In fact, using the depth-first pricing, the algorithm is not able to solve the larger instances within the imposed time-limit of 1 hour (see the instance with 500 nodes), because the pricing takes too much time and optimality cannot yet be proven because the initial upperbound (based on $K = L = \infty$) cannot be achieved for these instances. Subset arc branching appears to require the least amount of branching decisions of the various branching strategies, although the difference in performance is small, at least not on the instances tested. Often the optimal solution is already found in the root of the branch-and-bound tree.

Next, we perform experiments with instances constructed by the simulator with US population data described in Section 4. When we consider the unmodified version of the Saidman simulator to generate non-sparse pools, we obtain results similar to those in Abraham et al. (2007). However, as indicated by Ashlagi and Roth (2012), in practice pools turn out to become increasingly sparse. Therefore we directly proceed and report results for the modified version of the Saidman simulator that takes this higher degree of sparsity into account. In this case, we generate various sparse instances up to 1,000 nodes. The cycle and chain length limit is set to either $K = L = 3$ (short cycles and chains) or to $K = 4$ and $L = 6$ (long cycles and chains).

Table 5 summarizes the average performance characteristics over the generated instances. The columns in Table 5 are similar to the columns in Table 4, except that now, as not all versions of the algorithm are able to solve all the sparse instances, the percentage of solved instances is reported in the last column.

The findings reported in Table 5 for sparse instances generated with US data are in line with the findings reported in Table 4 for instances generated with Dutch data. When we allow only short cycles and chains the polynomial pricing algorithm and the depth-first pricing algorithm perform almost identically.

Pool size	Total time (s)	LP time (s)	IP time (s)	# nodes proc. / # nodes	Pricing time (s)
Depth first pricing with cycle branching, $K = 3, L = 3$					
10	.61	.04	.00	1 / 1	.00
20	.21	.04	.00	1 / 1	.00
50	1.11	.26	.00	7 / 13	.00
100	.78	.09	.28	1 / 1	.06
200	1.81	.52	.47	1 / 1	.36
500	54.92	32.59	.47	28 / 55	12.17
Depth first pricing with cycle brancing, $K = 4, L = 6$					
10	.28	.05	.00	2 / 3	.00
20	.83	.17	.00	5 / 9	.00
50	.42	.13	.00	2 / 3	.00
100	7.52	.14	.27	1 / 1	6.81
200	730.65	9.30	.20	16 / 31	713.17
500	>3600	-	-	-	>3600
Polynomial pricing with arc branching, $K = 3, L = 3$					
10	.33	.03	.00	1/1	.00
20	.20	.03	.00	1/1	.00
50	2.97	.91	.00	20/39	.00
100	.80	.09	.25	1/1	0.13
200	1.58	.17	.30	1/1	0.64
500	21.98	1.09	.69	1/1	18.08
Polynomial pricing with arc branching, $K = 4, L = 6$					
10	.27	.06	.00	2 / 3	.00
20	.94	.19	.00	6 / 11	.00
50	1.61	.53	.00	10 / 19	.00
100	.83	.13	.31	1 / 1	.08
200	2.78	.67	.30	1 / 1	1.33
500	103.67	21.44	4.53	24 / 47	35.72
Polynomial pricing with subset arc branching, $K = 3, L = 3$					
10	.16	.05	.00	1 / 1	.00
20	.16	.03	.00	1 / 1	.00
50	.50	.13	.00	3 / 5	.00
100	.70	.08	.22	1 / 1	.11
200	1.59	.17	0.30	1 / 1	.64
500	22.06	1.08	0.67	1 / 1	18.13
Polynomial pricing with subset arc branching, $K = 4, L = 6$					
10	.42	.09	.00	3 / 5	.00
20	.91	.22	.00	6 / 11	.00
50	1.20	.39	.00	7 / 13	.00
100	.88	.14	.31	1 / 1	.08
200	2.78	.67	.30	1 / 1	1.31
500	95.02	15.59	7.64	12 / 23	35.61

Table 4: Average performance characteristics for various instances generated with historical data from the Dutch national kidney exchange program.

Pool size	Total time (s)	LP time (s)	IP time (s)	# nodes proc. / # nodes	Pricing time (s)	% Solved
Depth first pricing with cycle branching, $K = 3, L = 3$						
100	.57	.14	.18	1 / 1	.02	100
200	3.01	1.83	.11	1.4 / 1.6	.61	100
500	60.26	37.29	.34	1 / 1	21.64	100
1000	421.53	224.25	1.47	1.4 / 1.8	191.08	100
Depth first pricing with cycle brancing, $K = 4, L = 6$						
100	1.05	0.29	0.42	1 / 1	0.09	100
200	53.75	9.96	0.25	1.6 / 2	42.89	100
500	2528.83	280.28	1.02	2.8 / 4.6	2260.56	90
1000	¿3600	1252.52	0.00	1 / 1	2384.20	0
Polynomial pricing with arc branching, $K = 3, L = 3$						
100	0.47	0.08	0.03	1 / 1	0.19	100
200	3.57	0.25	0.14	1 / 1	2.79	100
500	64.48	2.67	0.60	1 / 1	60.33	100
1000	431.35	52.59	2.66	1 / 1	372.71	100
Polynomial pricing with arc branching, $K = 4, L = 6$						
100	0.66	0.09	0.12	1 / 1	0.24	100
200	13.29	1.19	0.40	1 / 1	11.25	100
500	36.71	7.12	1.99	1 / 1	26.68	100
1000	326.57	54.55	16.92	6 / 11	172.25	100
Polynomial pricing with subset arc branching, $K = 3, L = 3$						
100	0.47	0.08	0.03	1 / 1	0.19	100
200	3.58	0.26	0.14	1 / 1	2.78	100
500	64.41	2.66	0.61	1 / 1	60.27	100
1000	446.17	57.29	2.56	1 / 1	379.18	100
Polynomial pricing with with subset arc branching, $K = 4, L = 6$						
100	0.66	0.10	0.12	1 / 1	0.24	100
200	13.29	1.21	0.40	1 / 1	11.22	100
500	36.67	7.10	1.97	1 / 1	26.68	100
1000	417.10	74.49	12.71	14.5 / 28	174.17	100

Table 5: Average performance characteristics over 10 randomly generated instances generated with US population data. 1 % of the donors is unspecified. 50 % of the patients is highly sensitized.

However, when we allow long cycles and chains, in almost all instances the polynomial pricing algorithm performs much better than the depth-first pricing algorithm. In fact, when using depth-first pricing with cycle branching, many of the larger instances cannot be solved (this is the case for 40 percent of the instances with 500 nodes, and 100 percent of the instances with 1,000 nodes) while all of these instance can be solved within a couple of minutes when using polynomial pricing.

As before, many instances can be solved in the root of the branching tree, but, in total, branching is now required for more instances. When branching is required, arc branching appears to be more effective than subset arc branching

Pool size	% Instances for which upperbound is attainable
$K = 3, L = 3$	
100	30
200	0
500	0
1000	10
$K = 4, L = 6$	
100	70
200	20
500	100
1000	100

Table 6: Percent of instances for which the upperbound based on $K = L = \infty$ is attainable

as it leads to less branches on average (see the 1,000 node instances).

The reason that the algorithm using depth-first pricing and cycle branching has difficulty solving sparse instances, is that for these instances the upperbound for the maximum number of transplants due to Roth et al. (2007) based on $K = L = \infty$ is no longer as tight as in non-sparse pools. Table 6 displays the percentage of instances used in Table 5 for which this upperbound is attainable. As can be seen from the table, the upperbound can almost never be attained - not even for large instances - when only short cycles and chains are allowed. When long cycles and chains are allowed, however, the upperbound can almost always be attained. But allowing long cycles and chains is very detrimental to the run time of the depth-first pricing algorithm. When the upperbound due to Roth et al. (2007) cannot be achieved, optimality cannot be proven in an early stage, and many cycles and chains have to be considered before the algorithm can conclude that no cycle or chain with positive reduced cost exists. As this causes the depth-first algorithm to enumerate a very large number of cycles and chains (in order to prove optimality), it takes a very long time to solve the pricing problems. The polynomial pricing algorithms perform better as they do not need to consider as many cycles and chains.

Of practical importance, however, is not just the run time, but also the impact of applying the iterative branch-and-price algorithm. Figure 2 displays the long term effects of using this algorithm for multi-criteria kidney exchange using the Dutch criteria. In particular, the figure shows the relative difference to single-criterion kidney exchange on the total number of transplants, the average wait time, the number of highly sensitized patients (patients with PRA > 80) transplanted, and O patients transplanted in 30 Monte Carlo simulations. Results are shown for two types of policies: policies with domino paired donation (DPD) chains, of which the last donor in the chain donates to a patient on the deceased donor waitlist, and policies with non-simultaneous extended altruistic donor (NEAD) chains, of which the last donor in the chain becomes a 'bridge' donor who can start a new chain in a subsequent exchange. For both policies we compare the multi-criterion solution to the corresponding single-criterion solution.

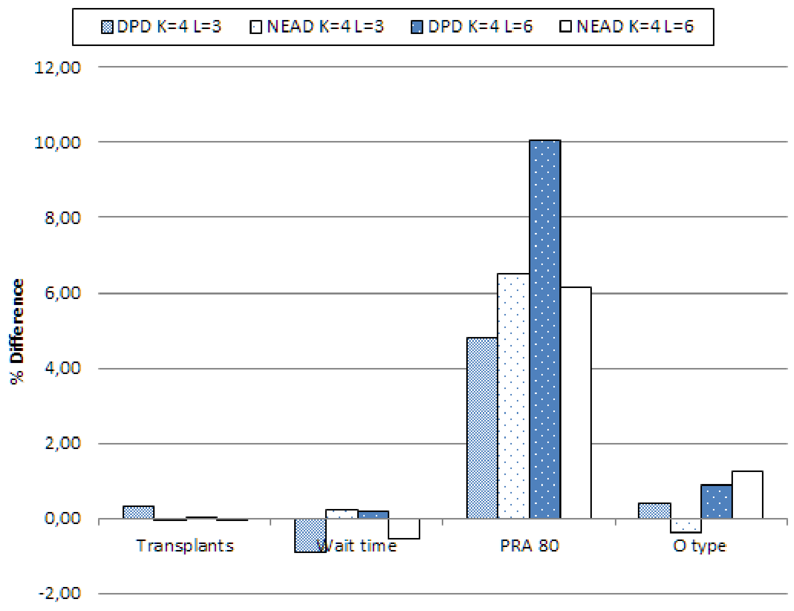

Figure 2: Long term impact of multi-criteria kidney exchange: relative difference to single-criterion policies for several criteria

While the difference in the total number of transplants and the average waiting time is negligible (which is as expected), the difference in terms of highly sensitized patients transplanted is significant, both statistically and practically (statistical significance is tested using the Sign test with $\alpha = 0.05$). This - normally disadvantaged - group can receive up to 10 percent more transplants when using the Dutch allocation criteria ($P < 0.001$). O type patients do not appear to benefit as much, but even for this group the difference is significant under the policies with $L = 6$ ($P = 0.0142$ for DPD and $P = 0.0339$ for NEAD). Most important though, is that the total allocation process satisfies the fairness requirements of international treaties (such as the European agreements mentioned in the introduction) as these are captured in the multiple criteria that are used to make the matching decisions.

6 Conclusions and further research

In this paper we have shown how to clear large multi-criteria kidney exchanges using a general and scalable exact algorithm. This is particularly important as, over the last years, kidney exchange has quickly increased as a modality for transplanting end stage renal disease patients with an incompatible living donor. Most kidney exchange programs not only seek to optimize the number of transplants, but also seek to guarantee a level of fairness, as prescribed in international treaties (e.g. Council of Europe (2002)). For this reason many programs use a set of multiple hierarchical optimization criteria. Using our algorithm, we can effectively deal with such criteria, even in large and sparse exchange pools with unspecified donors that now begin to arise in practice, whereas we show such pools to be problematic for existing algorithms.

To maximize the benefits from kidney exchange, the exchange should be coordinated at a national level and integrated with unspecified donation. However, participation barriers for transplant centers may prevent such nationally coordinated kidney exchange from being established. To make such coordination possible then, participation constraints must be included. Our algorithm can also deal effectively with such constraints.

Mathematically, the algorithm consists of an iterative branch-and-price procedure. By using a general but effective class of integer programming formulations we are able to optimally clear exchange pools with billions of cycles and chains within minutes. The key part of our algorithm is a polynomial pricing procedure for this class of formulations in combination with a branching strategy that branches on arcs or on subsets of arcs. These elements allow us to efficiently deal with long chains - which are an upcoming phenomenon in kidney exchange - which would not be possible with depth-first pricing techniques suggested in previous research.

On a practical level, we have shown through long term analysis that highly sensitized patients may on average receive up to 10 percent more transplants when using the hierarchical allocation criteria used in the Dutch kidney exchange program. Although for testing purposes the algorithm has been specifically implemented with the Dutch criteria, it easily generalizes to a wide variety of other criteria. One particular alternative criterion that might be thought of is, for example, the maximization of the number of expected life years gained. Optimizing with respect to that criterion might also open up the possibility of including compatible pairs in the kidney exchange program.

We hope our algorithm may serve as a reference solution framework for other researchers, so that solution methods and data can be shared, to the benefit of the patients suffering from end stage renal disease accross the globe.

References

Abraham, D, A Blum, T Sandholm. 2007. Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. *ACM EC* (07).

Ashlagi, I, DS Gilchrist, AE Roth, MA Rees. 2011. Nonsimultaneous chains and dominos in kidney paired donation - revisited. *American Journal of Transplantation* (11) 984–994.

Ashlagi, I, P Jaillet, VH Manshadi. 2013. Kidney exchange in dynamic sparse heterogenous pools. *Working paper* .

Ashlagi, I, A Roth. 2011. Free riding and participation in large scale, multi-hospital kidney exchange. *NBER paper no. w16720* .

Ashlagi, I, AE Roth. 2012. New challenges in multi-hospital kidney exchange. *American Economic Review, Papers and Proceedings* **102**(3) 354–359.

Barnhart, C, EL Johnson, GL Nemhauser, MWP Savelsbergh, PH Vance. 1998. Branch-and-price: Column generation for solving huge integer programs. *Operations Research* (46) 316–329.

Bellman, R. 1958. On a routing problem. *Quarterly of Applied Mathematics* (16) 87–90.

Constantino, M, X Klimentova, A Viana, A Rais. 2013. New insights on integer-programming models for the kidney exchange problem. *Preprint submitted to European Journal of Operational Research* .

Council of Europe. 2002. Additional protocol to the convention on human rights and biomedicine concerning transplantation of organs and tissues of human origin. *European Treaty Series* (186).

Delmonico, F, P Morrissey, G Lipkowitz, J Stoff, J Himmelfarb, W Harmon, M Pavlakis, H Mah, J Goguen, R Luskin, E Milford, G Basadonna, M Chobanian, B Bouthot, M Lorber, R Rohrer. 2004. Donor kidney exchanges. *Am J Transplant* (4) 1628–1634.

Ford, LR. 1956. Network flow theory. *Paper P-923, The RAND Corporation, Santa Monica, California* .

Glorie, KM, M de Klerk, APM Wagelmans, JJ van de Klundert, WC Zuidema, FHJ Claas, W Weimar. 2012. Unspecified donation in kidney exchange: when to end the chain. *Econometric Institute report* (2012-19).

Keizer, KM, M de Klerk, BJJM Haase-Kromwijk, W Weimar. 2005. The dutch algorithm for allocation in living donor kidney exchange. *Transplantation Proceedings* (37) 589–591.

Kim, BS, YS Kim, SI Kim, MS Kim, HY Lee, YL Kim, CD Kim, CW Yang, BS Choi, DJ Han, YS Kim, SJ Kim, HY Kim, DJ Kim. 2007. Outcome of multipair donor kidney exchange by a web-based algorithm. *Journal of the American Society of Nephrology* **18**(3) 1000–1006.

Klerk, M De, WM Van der Deijl, MD Witvliet, BJJM Haase-Kromwijk, FHJ Claas, W Weimar. 2010. The optimal chain length for kidney paired exchanges: an analysis of the dutch program. *Transplant Int.* (23) 1120–1125.

Klerk, M De, J Kal van Gestel, B Haase-Kromwijk, F Claas, W Weimar. 2011. Eight years of outcomes of the dutch living donor kidney exchange program. *Clinical Transplants 2011*. Terasaki Foundation Laboratory, Los Angeles, California.

Manlove, D, G O'Malley. 2012. Paired and altruistic kidney donation in the uk: Algorithms and experimentation. *Experimental Algorithms: Proceedings of the 11th International Symposium, SEA 2012, Bordeaux, France, June 7-9, 2012.* 271–282.

Montgomery, R, S Gentry, W Marks, D Warren, J Hiller, J Houp et al. 2006. Domino paired kidney donation: a strategy to make best use of live non-directed donation. *Lancet* (368) 419–421.

(NTS)., Nederlandse Transplantatie Stichting. ???? *http://www.transplantatiestichting.nl* .

Park, K, JI Moon, SI Kim, YS Kim. 1999. Exchange donor program in kidney transplantation. *Transplantation* **67**(2) 336–338.

Roth, AE, T Snmez, MU Unver. 2007. Efficient kidney exchange: Coincidence of wants in markets with compatibility-based preferences. *The American Economic Review* **97**(3) 828–851.

Roth, AE, T Snmez, MU Unver, FL Delmonico, SL Saidman. 2006. Utilizing list exchange and nondirected donation through chain kidney paired donations. *American Journal of Transplantation* (6) 2694–2705.

Saidman, S, A Roth, T Sonmez, U Unver, F Delmonico. 2006. Increasing the opportunity of live kidney donation by matching for two- and three-way exchanges. *Transplantation* **81**(5) 773–782.