Robust Models for the Kidney Exchange Problem

Kristiaan Glorie [a]

Margarida Carvalho [b,c]

Miguel Constantino [d]

Ana Viana [e,f]

Xenia Klimentova [e]

[a] Rotterdam School of Management, Erasmus University Rotterdam, Netherlands

[b] DIRO, Université de Montréal, Canada

[c] Canada Excellence Research Chair in Data Science for Real-Time Decision-Making

[d] Faculdade de Ciências da Universidade de Lisboa, Portugal

[e] INESC TEC, Porto, Portugal

[f] School of Engineering, Polytechnic of Porto, Portugal

glorie@ese.eur.nl
carvalho@iro.umontreal.ca
mfconstantino@fc.ul.pt
aviana@inesctec.pt
xenia.klimentova@inesctec.pt

Canada Excellence Research Chair in Data Science for Real-Time Decision-Making

Abstract: Kidney exchange programs enable transplants between incompatible donor-patient pairs: a set of pairs is chosen in such a way that each selected patient receives a kidney from a compatible donor from another pair in the set. The pairs are then notified and crossmatch tests are performed for the selected exchanges. These tests may reveal incompatibilities between pairs, preventing a planned exchange from proceeding. We study the case in which, if incompatibilities are discovered, a partaker has to withdraw and a new set of pairs may be selected. The new set should be as close as possible to the initial set in order to minimize the material and emotional costs of the changes. Various recourse policies that determine the allowed post-matching actions are proposed. For each recourse policy, a robust model is developed. Besides the development of a novel adjustable robust optimization model, our contribution includes techniques to solve exactly the optimization problems at hand. Computational results show that for instances of realistic size our models can be solved within run times that are acceptable for practice. More importantly, in a substantial number of instances it is possible to actually protect patients against failures that prevent them to undergo a transplant. In this regard, our algorithms may offer a significant improvement over current practice.

Keywords: Kidney exchange; Robust optimization; Integer programming.

Acknowledgments: This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project "mKEP - Models and optimisation algorithms for multicountry kidney exchange programs" (POCI-01-0145-FEDER-016677), by FCT project SFRH/BPD/101134/2014. The second author thanks the support of Institute for data valorisation (IVADO).

Robust Models for the Kidney Exchange Problem

Kristiaan Glorie [*] Margarida Carvalho [†‡] Miguel Constantino [§] Ana Viana [¶]
Xenia Klimentova [‖]

submitted: October, 2018

Abstract

Kidney exchange programs enable transplants between incompatible donor-patient pairs: a set of pairs is chosen in such a way that each selected patient receives a kidney from a compatible donor from another pair in the set. The pairs are then notified and crossmatch tests are performed for the selected exchanges. These tests may reveal incompatibilities between pairs, preventing a planned exchange from proceeding. We study the case in which, if incompatibilities are discovered, a partaker has to withdraw and a new set of pairs may be selected. The new set should be as close as possible to the initial set in order to minimize the material and emotional costs of the changes. Various recourse policies that determine the allowed post-matching actions are proposed. For each recourse policy, a robust model is developed. Besides the development of a novel adjustable robust optimization model, our contribution includes techniques to solve exactly the optimization problems at hand. Computational results show that for instances of realistic size our models can be solved within run times that are acceptable for practice. More importantly, in a substantial number of instances it is possible to actually protect patients against failures that prevent them to undergo a transplant. In this regard, our algorithms may offer a significant improvement over current practice.

Keywords— Kidney exchange, Robust optimization, Integer programming.

1 Introduction

Kidney exchange programs (KEP's) represent an alternative for patients suffering from end stage kidney failure to receive a transplant from a living donor (e.g., [17], [35]). If a patient has someone willing to donate him/her a kidney but patient and donor are physiologically incompatible, the pair can join a pool of incompatible pairs (the KEP) and if compatibility between donor and patient in different pairs is found, an exchange is allowed. For the simplest exchange, involving only two pairs P_1 and P_2, the patient from P_1 would receive a kidney from the donor in P_2 and vice-versa. The concept can be extended to *k-cycle*, when k pairs are involved in the transplants. The solution of a KEP is a set of disjoint exchange cycles that optimize a given objective. Figure 1 represents a kidney exchange pool with six pairs. Nodes represent incompatible pairs and arcs indicate compatibility between pairs. The solution that leads to maximum number of transplants is represented by cycles (1, 2, 3) and (4, 6) – arcs associated to the planned transplants are represented in bold.

The example in Figure 1 represents a planned solution (set of kidney transplants) that may totally or partially fail to go forward if a crossmatch test, that is performed after the pairs that are selected for transplant, is positive (see [19, 17, 29]). In such cases, pre-planned transplants involving those pairs are

[*]Rotterdam School of Management, Erasmus University Rotterdam, Netherlands.
[†]DIRO, Université de Montréal, Montréal, Canada.
[‡]Canada Excellence Research Chair in Data Science for Real-Time Decision-Making.
[§]Faculdade de Ciências da Universidade de Lisboa, Lisboa, Portugal.
[¶]INESC TEC and School of Engineering, Polytechnic of Porto, Porto, Portugal.
[‖]INESC TEC, Porto, Portugal.

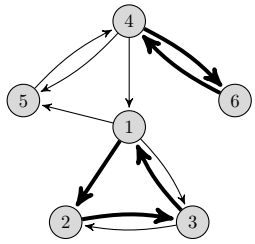


Figure 1: A kidney exchange program pool and respective solution (in thick lines).

cancelled and, if no recourse actions are implemented, the number of proposed and effective transplants may substantially differ.

Most works related to the optimisation of KEPs that considers data uncertainty, associate probabilities of failure to vertices and arcs of the graph. In [5, 15, 23, 33, 37, 31], the authors consider the expected number of transactions that can go forward by associating probabilities to node and arc failure. However, considering the expectation is not always tractable or desirable. For instance, in kidney exchanges there is a class of patients who are highly sensitized, which means that these patients are compatible with only a very small fraction of kidney donors. The rare matching opportunities that exist for these patients should be protected against failure. Failure to match highly sensitized patients has led to the accumulation of these patients in kidney exchange pools and in substantially longer waiting times and higher mortality for these patients [3]. Furthermore, assessing the probability of failure of each node/arc is in many cases impossible due to the lack of reliable data.

In this research we focus on robust approaches (see for example [6]) that allow us to specify the desired level of protection from uncertainty, namely, one can decide on the polytope of feasible attacks. A substantial advantage of using robust optimization is that it requires no assumptions on the underlying probability distribution. We consider various recourse policies that determine the allowed actions after an initial subset of transplants is proposed for verification. In our first policy, called *simple recourse*, we take into account costs (or missed gains) for failing transactions. Although this policy does not allow failing transaction cycles to be recovered, it does allow better decisions to be made regarding the set of transactions that is initially proposed because the possibility of failure is taken into account. In the second policy, called *back-arcs recourse*, we allow part of a failing transaction cycle to be recovered if the remaining participants in the cycle can be transplanted among themselves. In our last policy, *full recourse*, we allow for a complete recovery of the initial solution using alternative transactions. We develop robust models for each of the recourse policies and propose techniques to solve exactly the optimization problems at hand. Computational results show that for instances of realistic size our robust models can be solved within run times that are acceptable in practice. More importantly, in a substantial number of instances, it is possible to actually protect patients against failures that prevent them to undergo a transplant.

The remainder of this paper is organized as follows. Section 2 gives a brief overview of the kidney exchange problem. Section 3 provides a mathematical description of the robust exchange problem. It first presents a general model for market uncertainty and then details each policy: the simple recourse, back-arcs recourse, and full recourse. Section 4 describes our theoretical results for each of these recourse policies. Section 5 considers a method to refine the robust solution by embedding the robust optimization criteria in a hierarchical set of criteria. Section 6 then provides computational results. Finally, Section 7 concludes.

2 Kidney exchange programs: an overview

Kidney exchange programs have received substantial attention in recent years as they represent an additional possibility for patients suffering from end stage kidney disease of being transplanted with a healthy kidney. National programs have been set up in several countries (e.g. the Netherlands [17], the US [38], the United Kingdom [35]), under different regulations. Programs also differ on the composition of the pool, as well as on the objective(s) to optimize. Initially only incompatible patient-donor pairs participated in KEP's but

nowadays compatible pairs may also be included. These pairs are given an incentive to participate in the KEP (e.g., the patient may receive a more suitable kidney than if direct transplant from his/her donor was done) and benefit the program if they allow more transplants to be triggered. Furthermore, KEP's can have altruistic donors (i.e. donors that do not have an associated patient): an altruistic donor donates to a patient in the KEP, the patient's associated donor donates to another patient and so forth. The last donor in the chain either donates to a patient in the deceased donors' waiting list, or acts as a "bridge donor" for future matches [16]. Usually chains are also assigned a maximum size, l. In terms of objectives, the maximization of the number of transplants seems to be the most commonly accepted evaluation criteria considered. But other criteria such as maximizing the number of blood type identical matches (to maximize the likelihood of O patients receiving a kidney and to help overcome their disadvantage), or prioritizing allocations based on the number of involved recipients with a low match probability, are also studied [27].

The clearing problem associated to a KEP can be modelled through Integer Programming. It was formulated as a cycle packing problem in a graph in [1] and the authors provided a branch-and-price algorithm. Their approach worked well for cycles and chains involving up to three pairs. [27] showed how the pricing problem could be solved efficiently in the cycle and chain length, thereby allowing the branch-and-price approach to scale better to longer cycles and chains. [16] proposed and analyzed the performance of alternative compact edge formulations. The formulations can be adapted to incorporate problem variants such as the possibility of a patient having multiple donors, or inclusion of altruistic donors. More recently, [22] presented two new compact IP formulations. Furthermore, they showed that one of those formulations has a linear programming relaxation that is exactly as tight as the previous tightest formulation known – the *cycle* formulation.

All the above referred work considers that the problem is deterministic i.e. that no pairs will dropout of the program in between pair matching and the actual transplant. However, that is not the case in practice: pairs can actually withdraw from the program and it can also happen that final crossmatch tests detect incompatibilities between pre-selected pairs. In both cases the transplants involving those pairs cannot proceed and the actual number of transplants performed differs from the planned number.

Failure was first considered in [5], who heuristically solved the online clearing problem using scenario sampling to minimize regret over several future scenarios. In [20], they presented an alternative heuristic learning approach to deal with uncertain future scenarios. Their approach relied on using weighted myopia. The same authors optimized the set of proposed transactions considering probabilistic failures, but allowed no recourse to recover trading cycles [21]. Their approach resembles the simple recourse proposed in this work in a probabilistic setting. [36] and [30] considered failures in a bilateral exchange setting where transactions that verify positively must be executed.

Possibilities for recourse in KEPs were first considered in [35]. They introduced the notion of back-arcs recourse, and gave preference to cycles containing back-arcs in the clearing problem. Back-arcs recourse was also considered in [33] in a stochastic optimization setting where different probabilities of failure are assumed for each element. A stochastic model in which, upon failure, remaining pairs that have not been matched may (repeatedly) be subject to a new assignment was studied in [37]. In [31] the stochastic model is proposed considering simple and back-arc recourse possibilities. A bi-objective problem is studied with Conditional Value at Risk as a second criteria.

3 Robust optimization models for kidney exchange programs

Robust optimization was first introduced in [39]. Under the assumption of 'unknown but bounded' data, the goal was to optimize the objective value while guaranteeing feasibility with respect to all realizations of the data within the considered 'uncertainty set'. Because Soyster's approach tends to provide very conservative solutions, [34, 8, 9, 24, 25] developed new robust optimization frameworks for Integer Programming and Convex Programming that allow adjusting the size and shape of the uncertainty set to reach a balance between feasibility and the attainable objective value. Both static and dynamic approaches to robust optimization have been considered in [12]. In a static uncertain optimization problem, all decisions have to be made before the actual realizations of the parameters are known. In a dynamic uncertain optimization problem, some

decisions - the so-called 'recourse actions' - may be made after the parameters' values are known [7, 4, 14, 11]. Dynamic problems are referred to as two-stage or multi-stage problems, depending on the number of stages in which the decisions can be made. In [7] the authors show that two-stage robust linear programming is computationally intractable and propose a tractable alternative referred to as affinely adjustable robust linear programming. Affinely adjustable robustness requires the recourse decision variables to be an affine function of the realizations of the uncertain parameters.

The problem considered in this paper can be classified as a two-stage dynamic uncertain optimization problem. In contrast to the robust optimization approaches discussed above, our focus is not on maintaining feasibility in all scenarios, but instead on maximizing the gains of trade in the worst-case scenario in our uncertainty set. Exchanges that are infeasible after second stage recourse actions, are considered as lost exchanges. In this section we present a mathematical description of the robust kidney exchange problem and propose different recourse policies. More precisely, in Section 3.1 we introduce the general robust model, in Section 3.2 the recourse policies are described, and in Section 3.3, the robust solution before failure is compared with the optimal solution for the deterministic case.

3.1 Robust exchange model

Let $D = (V, A)$ denote an unreliable graph where the node set V is composed of two subsets: the set of incompatible pairs P, and altruistic donors N, $V = P \cup N$. The arc set A represents compatibilities. A *length k cycle* is a cycle $(v_1, \ldots, v_k)$ in graph D such that $\{v_1, \ldots, v_k\} \subseteq P$. A *length l altruistic donor chain* is a path $(v_1, \ldots, v_l)$ in graph D such that $v_1 \in N$ and $\{v_2, \ldots, v_l\} \subseteq P$. A solution to the clearing problem corresponds to a set of vertex disjoint cycles and chains in D with length at most k and l, respectively. For easiness of representation, consider that maximum cycle and chain length are equal, with value k, and denote by $\mathcal{C}$ the set of all cycles and chains in D with cardinality at most k. For any cycle or chain $c \in \mathcal{C}$ we denote by $V(c)$ and $A(c)$ a set of vertices and arcs, respectively, included in c.

As mentioned above, the cycle packing formulation was one of the most effective models for solving the problem with no uncertainty considered. Define the decision variables x_c as

$$x_c = \begin{cases} 1 & \text{if cycle/chain } c \in \mathcal{C} \text{ is selected in the planned solution,} \\ 0 & \text{otherwise.} \end{cases}$$

The cycle packing formulation is written as follows:

$$\max_x \sum_{x \in \mathcal{C}} w_c x_c \tag{1}$$

$$\text{s.t.} \sum_{c \,:\, v \in V(c)} x_c \leq 1 \quad \forall\, v \in V \tag{2}$$

$$x \in \{0,1\}^{|\mathcal{C}|} \tag{3}$$

where $w_c = \sum_{a \in A(c)} w_a$ denotes the benefit associated to executing a cycle or chain $c \in \mathcal{C}$, and w_a is a weight associated to arc $a \in A$. If $w_c = |P \cap V(c)|$ the objective is the maximization of the number of transplants. In what follows we will refer to the problem (1)-(3) as deterministic problem, and denote by Z^D its optimal value.

In order to address uncertainty, consider $\mathcal{U} \subseteq 2^{V \cup A}$ the collection of possible *scenarios* of ultimately available nodes and arcs, $u = V^u \cup A^u$, $V^u \subseteq V$, $A^u \subseteq A$, i.e. each scenario defines the transplants that can go forward after verification. A scenario set can be characterized, for example, by homogeneous failure:

Definition 1 *The setting in which at most p% of the nodes and arcs can fail is called* homogeneous*e. This corresponds to the uncertainty set* $\mathcal{U} := \{u = V^u \cup A^u \subseteq V \cup A : |V^u| + |A^u| \geq (|V| + |A|)(100 - p)\%\}$.

Furthermore, under a given scenario $u \in \mathcal{U}$, define a vector $\zeta^u = \{0,1\}^{|\mathcal{C}|}$, that indicates the availability of each cycle and chain under the scenario as follows:

$$\zeta_c^u = \begin{cases} 1, & \text{if } V(c) \subseteq V^u \text{ and } A(c) \subseteq A^u, \\ 0 & \text{otherwise.} \end{cases}$$

To model the robust exchange problem, that determines a solution to the clearing problem that is robust against uncertainty, Consider a *recourse function* $R(x, u)$ that specifies the objective value attained under the recourse policy for the planned solution $x = (x_1, \ldots, x_{|\mathcal{C}|})$ and the scenario $u \in \mathcal{U}$. The robust exchange problem is given by:

$$\max_{x} \min_{u \in \mathcal{U}} \quad R(x, u) \tag{4}$$

$$\text{s.t.} \quad \sum_{c \,:\, v \in V(c)} x_c \leq 1 \quad \forall\, v \in V \tag{5}$$

$$x \in \{0, 1\}^{|\mathcal{C}|} \tag{6}$$

The objective (4) is to maximize some objective value, as determined by the selected recourse policy, in the worst case scenario (defined by the internal minimization problem on set of scenarios $\mathcal{U}$). The packing constraints (5) guarantee that each pair can be involved in at most one cycle or chain.

One advantage of the robust solution approach is that, opposed to the stochastic optimization, the probability distribution for the uncertainty scenarios, which might be very difficult, or even impossible, to achieve, is not needed.

3.2 Recourse policies for kidney exchange programs

There are several kidney exchange programs that consider the possibility of recourse in case of failure of nodes or arcs between the time that elapses from pair matching and actual transplant. For example, in UK [35] they test a cycle together with its back-arcs, if they exist. The recourse action on back-arcs is performed in case of failure of elements of the main cycle. In the Dutch program [17] the cycles are tested sequentially. If no failure identified, the cycle is fixed, otherwise entire solution is re-optimized.

Although there are ethical reasons to justify such behaviour (a topic that goes beyond the scope of this paper), it is also commonly agreed among the peers that recourse might be considered as far as it does not affect pairs that were selected and could proceed to transplant. This motivates for the simple and back-arc recourse discussed in this paper. The last recourse policy discussed – full recourse – still captures the main concerns of not significantly affecting previously selected pairs. Although it allows for complete recovery of the planned solution using alternative exchanges, at the same time it maximizes the number of pairs that are both in the original planned solution and the one obtained after recourse.

3.2.1 Simple recourse

In this policy we take into account costs (or missed gains) for planned transplants that do not proceed: the alternative solution that may be selected after verifying the planned transactions consists only of the cycles and chains in the planned solution for which all nodes and arcs are available. Although it does not allow failing cycles or chains to be recovered, it ostensibly allows to make better decisions regarding the set of transplants that is initially proposed because the possibility of failure is explicitly taken into account.

The simple recourse function is defined as:

$$R_{\text{Simple}}(x, u) := \sum_{c \in \mathcal{C}} w_c \zeta_c^u x_c \tag{7}$$

If the weights $w_c = |P \cap V(C)|$ for each cycle or chain $c \in \mathcal{C}$ (i.e. number of transplants, associated to executing a cycle or chain), then $R_{Simple}(x, u)$ equals the number of arcs belonging to cycles and chains that are in the planned solution x and are feasible under scenario u.

The simple recourse may appear to be very restrictive as it does not allow failing cycles or chains to be recovered, however this restrictiveness corresponds to current practice in some kidney exchange programs as mentioned previously in this section. By taking the possibility of failure already into account in the primary decision stage, the simple recourse model allows to make better decisions even for those programs. Moreover,

the model can be easily adapted to be used as a selection criteria among the multiple optimal solutions of the deterministic program.

3.2.2 Back-arcs recourse

Under this policy we allow part of a failing cycle or chain to be recovered if the remaining participants in the cycle or chain can be transplanted among themselves.

Definition 2 Let c be a cycle or chain in graph D. An arc $a \in A : a = (i, j)$ is called a *back-arc for* c if $i, j \in V(c)$, but $a \notin A(c)$.

To formally define this recourse policy we consider additional decision variables, x_c^u that will identify the recourse decision under scenario u:

$$x_c^u = \begin{cases} 1 & \text{if cycle or chain } c \in \mathcal{C} \text{ is selected in the recourse solution} \\ & \text{under scenario } u \in \mathcal{U}, \\ 0 & \text{otherwise.} \end{cases}$$

The back-arcs recourse function can be written as follows:

$$R_{\text{Back-arcs}}(x, u) := \max_{x^u} \sum_{c \in \mathcal{C}} w_c x_c^u \tag{8}$$

$$\sum_{c' : V(c') \subseteq V(c)} x_{c'}^u \leq x_c \quad \forall c \in \mathcal{C} \tag{9}$$

$$x_c^u \leq \zeta_c^u \quad \forall c \in \mathcal{C}, \tag{10}$$

$$x^u \in \{0, 1\}^{|\mathcal{C}|} \tag{11}$$

The recourse objective (8) maximizes the benefit of the exchanges selected in the final solution given a specific scenario $u \in \mathcal{U}$. Constraints (10) ensure that only available cycles or chains can be selected, while constraints (9) guarantee that only cycles or chains c' embedded in cycle c will be chosen.

This policy is relevant as it allows a recovery stage, but does not include new participating nodes, reducing possible node failures in the recovery plan.

3.2.3 Full recourse

The last recourse policy we consider, called full recourse, allows for a complete recovery of the planned solution using alternative pairs. The aim is to determine a planned and recourse solution such that the number of nodes in the intersection of both solutions is maximized.

Using the same notation of previous models, the full recourse is defined as follows:

$$R_{\text{Full}}(x, u) := \max_{x^u} \sum_{c \in \mathcal{C}} \left(\sum_{v \in V(c)} \sum_{c' : v \in V(c')} x_{c'} \right) x_c^u \tag{12}$$

$$\text{s.t.} \sum_{c : v \in c} x_c^u \leq 1 \quad \forall\, v \in V^u \tag{13}$$

$$x_c^u \leq \zeta_c^u, \quad \forall c \in \mathcal{C}, \tag{14}$$

$$x^u \in \{0, 1\}^{|\mathcal{C}|} \tag{15}$$

In the recourse objective (12), the coefficient of x_c^u is the number of nodes from cycle c in the planned solution x. Thus, the objective (12) maximizes the number of nodes selected in both the initial and the final solution given a specific scenario $u \in \mathcal{U}$. Constraints (13) ensure that nodes can be selected at most once in the final solution, while constraints (14) guarantee that only cycles or chains available under u can be selected.

3.3 Optimality of the robust solution in the deterministic problem

To apply the robust solution in practice it is important to address the question, whether the robust solution is optimal in the deterministic problem (1)–(3), i.e. the objective values of the recourse solution x^* in the deterministic problem, denoted by $Z^D(x^*)$, is equal to the Z^D. In fact, this is always true if cycles and chains length limitation is $k = 2$ and their weight is 2. Following the results in [10], any robust solution (matching) can be augmented to be a maximum matching enabling us to conclude:

Lemma 1 *If $k = 2$, an optimal solution x^* for the simple, back-arc and full recourse problem can be augmented to be a maximum matching (without uncovering nodes present in x^*), i.e. there exists an optimal solution x^* such that $Z^D(x^*) = Z^D$.*

For $k = 3$ and bigger, the optimal robust solution may fail to be optimal to the deterministic case as examplified below.

Example 1 (Simple and Back-arc recourse case) Consider the instance represented in Figure 2, $k = 3$, homogeneous failure setting with at most 4 failing nodes or arcs. The optimal solution x^* for the simple and back-arcs recourse robust problem is to have as initial exchange plan the cycles in green (cycles (2,4), (3,5), (8,10) and (9,10)), with $Z^D(x^*) = 8$. At least 2 nodes are always maintained after an attack, since such solution has 5 cycles of size 2, i.e. $Z^* = 2$.

On the other hand, for the deterministic case, the optimal solution are the red cycles ((1,3,2), (4,5,6), (7,8,9) and (10,11,12)) with optimal value $Z^D = 12$, where under the worst failure all 4 cycles are attacked and no node can be recovered, even under the back-arc recourse.

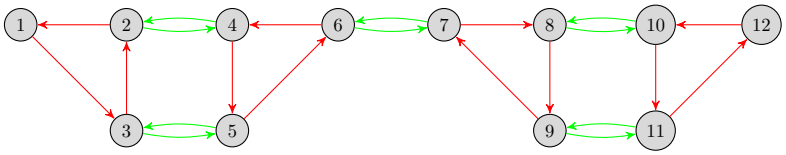


Figure 2: Compatibility graph.

Example 2 (Full recourse case) Consider the instance represented in Figure 3, $k = 3$, homogeneous failure and maximum number of failures is 2. The optimal solution x^* for the full recourse problem is to have as initial exchange plan the green cycles ((1,2) and (3,4)) with $Z^D(x^*) = 4$, a number of covered nodes before failure. At least 2 nodes are always recovered after an attack, since there is a cycle of length 2 between any of the 4 covered nodes, i.e. $Z^* = 2$.

On the other hand, for the deterministic case, the optimal solution are the red cycles ((2,6,5) and (3,7,8)) with $Z^D = 6$ covered nodes before failure; the worst case attack eliminates node 2 and 3, and thus, after failure, no node from the deterministic solution can be recovered.

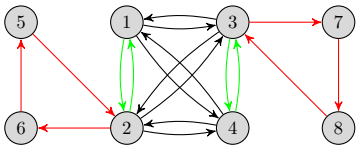


Figure 3: Compatibility graph.

Nevertheless, our computational results will show that the optimal robust solution is almost always also optimal for the deterministic problem.

4 Solving the robust exchange problem

In this section, we present a general approach for solving the robust kidney exchange problem. The proposed *delayed scenario generation method* is based on the algorithm developed by [2] for "defender-attacker-defender" models, and essentially differs it in the step requiring generation of the solution corresponding to the scenario with maximal "damage", the so-called adversary's problem. In [2], a Benders decomposition method is applied, while in our methods, in case of simple and back-arc recourse the adversary's problem is modeled by a Mixed Integer Program (MIP). For the case of full recourse we develop a branch-and-bound algorithm taking advantage of some structural results associated to our special setting. Scenario generation has been considered in stochastic programming (see e.g. [13]), but to the extend of our knowledge in has not been used before in adjustable robust optimization.

Moreover, for the simple and back-arc recourse, by use of the properties of the problem, we were able to formulate the robust problem under this policies as a MIP.

4.1 Delayed scenario generation

Since the set of scenarios $\mathcal{U}$ is finite, the robust exchange problem (4)-(6) can be alternatively formulated as the following large program.

$$
\begin{aligned}
RE(\mathcal{U}) := \quad & \max_{Z,x} \quad Z && && (16)\\
& \text{s.t.} \quad Z \leq R(x,u) && \forall\, u \in \mathcal{U} && (17)\\
& \qquad \sum_{c:v\in V(c)} x_c \leq 1 && \forall\, v \in V\\
& \qquad Z \in \mathbb{R}^+ && && (18)\\
& \qquad x \in \{0,1\}^{|\mathcal{C}|}
\end{aligned}
$$

Here, Z is an auxiliary decision variable and constraints (17) specify that the worst case is taken with respect to the uncertainty set $\mathcal{U}$.

The first difficulty in solving the robust exchange problem, regardless of the form of recourse, is that the number of scenarios (and hence the number of constraints (17) in the formulation) is typically prohibitively large to solve the problem directly as a mixed integer program. To try to overcome this difficulty we develop the delayed scenario generation algorithm (see Algorithm 1). The main idea of the algorithm is to start with a small set of scenarios and to generate additional scenarios only when required, i.e. when the corresponding constraint (17) is violated. We will see that for full recourse (see section 4.4) the generation of a new scenario implies simultaneous generation of a row and a column in the problem (16)-(18).

Algorithm 1 initializes the set of scenarios $\bar{\mathcal{U}}$ with the scenario where all vertices and arcs are available (Step 1). Then, in Step 2, $RE(\bar{\mathcal{U}})$ is solved. For the obtained solution, in Step 3 it is verified if there is a scenario capable of decreasing the number of exchanges. If there is, in Step 4 that scenario is added to the set $\bar{\mathcal{U}}$. Otherwise, the optimal solution is returned.

Algorithm 1 Delayed scenario generation algorithm

1: Let $\bar{\mathcal{U}} := \{N \cup A\}$.
2: Solve $RE(\bar{\mathcal{U}})$, let (Z^*, x^*) be the optimal solution.
3: **if** $\exists u \in \mathcal{U}\backslash\bar{\mathcal{U}}$ such that $RE(\bar{\mathcal{U}} \cup \{u\}) < RE(\bar{\mathcal{U}})$ **then**
4: Set $\bar{\mathcal{U}} \leftarrow \bar{\mathcal{U}} \cup \{u\}$ and go to **Step 2**.
5: **else**
6: Return (Z^*, x^*).
7: **end if**

Due to the finiteness of $\mathcal{U}$, Algorithm 1 terminates in a finite number of iterations. This allows us to claim the following.

Proposition 1 *Algorithm 1 returns the optimal solution to the robust exchange problem $RE(\mathcal{U})$.*

Proof. Indeed, due to Step 3, there is no $u \in \mathcal{U}\backslash\bar{\mathcal{U}}$ such that $RE(\mathcal{U} \cup \{u\}) < RE(\mathcal{U})$. Hence, all constraints (17) are satisfied by the returned solution x^* and that solution is optimal to $RE(\mathcal{U})$. □

In order to verify Step 3 of the algorithm, given a planned solution x, we can compute the worst case scenario by solving the following *adversary's problem*:

$$A(x) := \min_{u \in \mathcal{U}} R(x, u) \tag{19}$$

The objective of the problem minimizes the benefits after a recourse decision is taken.

As Step 3 of the algorithm is equivalent to solving (19), Proposition 2 below holds.

Proposition 2 *Let (Z^*, x^*) denote an optimal solution to the robust exchange problem $RE(\bar{\mathcal{U}})$. Furthermore, let u^* denote an optimal solution of the adversary's problem $A(x^*)$ with objective value z^*. There exists no scenario $u \in \mathcal{U}\backslash\bar{\mathcal{U}}$ such that $RE(\bar{\mathcal{U}} \cup \{u\}) < RE(\bar{\mathcal{U}})$ if and only if $z^* \geq Z^*$.*

Proof. If there is no scenario $u \in \mathcal{U}\backslash\bar{\mathcal{U}}$ such that $RE(\bar{\mathcal{U}} \cup \{u\}) < RE(\bar{\mathcal{U}})$, by Proposition 1 we may conclude that (Z^*, x^*) is the optimal solution of the problem $RE(\mathcal{U})$. Hence $Z^* \leq R(x^*, u)\ \forall u \in \mathcal{U}$ and we establish that $Z^* \leq z*$ (as $z^* = \min_{u \in \mathcal{U}} R(x^*, u)$.

In another way, if $z^* \geq Z^*$, then we may conclude that all the constraints (17) are satisfied and x^* is optimal solution. Hence there can not exist scenario $u \in \mathcal{U}\backslash\bar{\mathcal{U}}$ such that $RE(\bar{\mathcal{U}} \cup \{u\}) < RE(\bar{\mathcal{U}})$. □

In the following subsections we will apply this methodology for the considered recourse functions. For the sake of simplicity and due to the fact that there is no relevant public data that allows us to build the constraints defining the uncertainty scenarios, we restrict our analyses to homogeneous failure (i.e. the uncertainty set corresponds to the cases where at most $p\%$ of the nodes and arcs fail). For the cases of simple and back-arc recourse this assumption will allow us to propose an alternative way of solving the problem by reducing it to an equivalent mixed integer program.

Nevertheless, the proposed methodology can be generalized for the cases of non-homogeneous failure or cycles/chains of bigger size, if the the set of scenarios can be represented by a system of linear constraints and function $R(x, u)$ is also linear. In this case, both for simple and back-arc recourse the adversary's problem is modeled as a mixed integer program, and can be solved by standard optimization solvers. For the full recourse problem the algorithm proposed in section 4.4.1 can be directly applied.

4.2 Simple recourse

To adapt the proposed methodology for solving the robust exchange problem with simple recourse, we start by presenting the corresponding adversary's problem:

$$A_{\text{Simple}}(x) := \min_{u \in \mathcal{U}} \sum_{c \in \mathcal{C}} w_c \zeta_c^u x_c. \tag{20}$$

The robust exchange problem for the simple recourse is written as follows:

$$RE_{Simple}(\mathcal{U}) := \max_{Z, x} \quad Z \tag{21}$$

$$\text{s.t.} \quad Z \leq \sum_{c \in \mathcal{C}} w_c \zeta_c^u x_c \quad \forall\ u \in \mathcal{U} \tag{22}$$

$$\sum_{c : v \in V(c)} x_c \leq 1 \quad \forall\ v \in V$$

$$Z \in \mathbb{R}_+ \tag{23}$$

$$x \in \{0,1\}^{|\mathcal{C}|}$$

When Algorithm 1 is applied to this problem, the adversary's problem (20) is solved in Step 3 until the condition of Proposition 2 is satisfied. For homogeneous failure setting, let us denote by let $b = \lfloor p\%(|N|+|A|)\rfloor$ an uncertainty 'budget' (number of vertices and arcs to be attacked). Then a straightforward strategy for the adversary is to cancel the b most valuable cycles and chains, as a single arc or node failure completely cancels them. In our implementation, from the set of most valuable cycles we create a scenario which interdicts the vertices that participates in the largest number of cycles or chains.

4.2.1 MIP model for solving the simple recourse robust KEP

Considering the observation above, we can simplify the adversary's problem by replacing node and arc failure with cycle and chain failure. We do so by transforming a vector ζ^u into a decision variable vector ζ, such that $\zeta_c = 1$, if cycle or chain c is chosen to be attacked, 0 otherwise for $c \in \mathcal{C}$, and rewrite the adversary's problem as:

$$\begin{aligned}
A_{\text{Simple}}(x) := \quad \min_{\zeta} \quad & \sum_{c\in\mathcal{C}} w_c x_c \zeta_c && (24)\\
\text{s.t.} \quad & \sum_{c\in\mathcal{C}} \zeta_c \geq |\mathcal{C}| - b && (25)\\
& \zeta_c \leq 1 \quad \forall c \in \mathcal{C} && (26)\\
& \zeta \in \{0,1\}^{|\mathcal{C}|} && (27)
\end{aligned}$$

By use of sufficient conditions from [40] it can be shown that the constraint matrix of the $A_{\text{Simple}}(x)$ is totally unimodular (see proof in Appendix 8.1). As a result, one can conclude that every extreme point of the feasible region is integral and constraints (27) can be relaxed.

Using strong duality on the relaxation of the adversary's problem (24) – (26) and letting r denote the dual of constraint (25) and h_c, $c \in \mathcal{C}$ be the dual of the unit upper bounds (26), we can obtain an equivalent mixed integer programming formulation of the robust exchange problem under simple recourse as follows:

$$\begin{aligned}
MIP_{Simple} := \quad \max_{x,v} \quad & (|\mathcal{C}| - b)\, r - \sum_{c\in\mathcal{C}} h_c \\
\text{s.t.} \quad & \sum_{c\in\mathcal{C}(k):n\in c} x_c \leq 1 \qquad \forall v \in V \\
& r - h_c \leq w_c x_c \qquad \forall c \in \mathcal{C} \\
& x_c \in \{0,1\} \qquad \forall c \in \mathcal{C} \\
& r \geq 0 \\
& h_c \geq 0 \qquad \forall c \in \mathcal{C}
\end{aligned}$$

Although the above model is still a mixed integer program, its size is smaller than that of formulation RE_{Simple}.

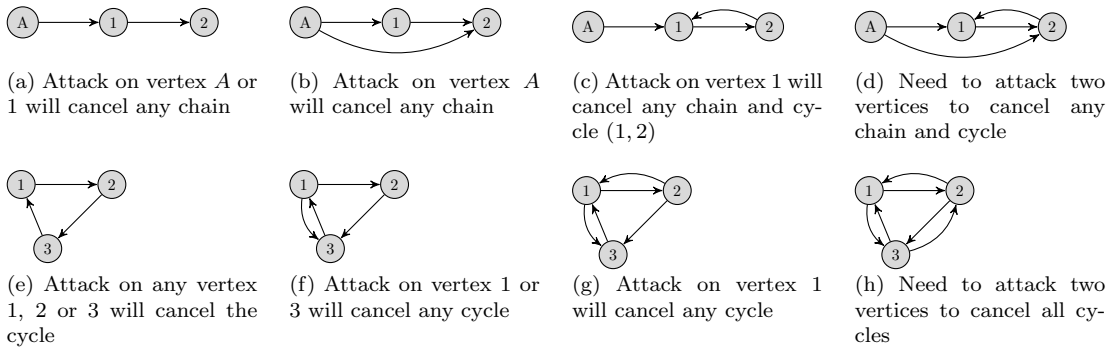
4.3 Back-arcs recourse

In this section we build solution approaches to the robust exchange problem with back-arcs recourse. Similarly to the analysis for the simple recourse, we begin by considering the adversary's problem:

$$A_{\text{Back-arcs}}(x) := \min_{u \in \mathcal{U}} R_{\text{Back-arcs}}(x, u)$$

where $R_{\text{Back-arcs}}(x, u)$ is defined in (8)-(11)

Since the back-arcs recourse function, $R_{\text{Back-arcs}}(x, u)$ is a maximization problem, the adversary's problem is, in the general case, a non-linear optimization problem. However, when $k = 3$ and considering $w_c = |P \cap V(c)|$ for all $c \in \mathcal{C}$ (i.e. we want to maximize the number of transplants) the adversary's problem can be linearized. By definition, back-arcs do not exist for cycles of length 2 and back-arcs recourse is meaningless for chains of length 2, then we only have to consider recourse actions for cycles and chains of length 3. There are four possible configurations for back-arc reaction in chains of size 3 (see Figure 4 (a)–(d)), and four possible configurations for cycles of the same size (see Figure 4 (e)–(h)). Note that, in the absence of failure, a length 3 chain involves 2 pairs from P (set of incompatible pairs), i.e. $w_c = 2$, and a length 3 cycle involves 3 pairs from P, $w_c = 3$.



(a) Attack on vertex A or 1 will cancel any chain

(b) Attack on vertex A will cancel any chain

(c) Attack on vertex 1 will cancel any chain and cycle $(1, 2)$

(d) Need to attack two vertices to cancel any chain and cycle

(e) Attack on any vertex 1, 2 or 3 will cancel the cycle

(f) Attack on vertex 1 or 3 will cancel any cycle

(g) Attack on vertex 1 will cancel any cycle

(h) Need to attack two vertices to cancel all cycles

Figure 4: Configurations of back-arcs for cycles and chains of length 3.

In the first three cases for chains (Figure 4, (a)-(c)) and cycles (Figure 4, (e)-(g)) a single node failure is sufficient to forbid all nodes in the chain to be recovered, but in the last cases (Figure 4, (d) and (h)), two failures are required. According to this reasoning, next we reformulate the adversary's decision variables to obtain a linearization of the back-arcs recourse function $R_{\text{Back-arcs}}(x, u)$.

Let $\bar{\mathcal{C}} \subset \mathcal{C}$ be as set of chains and cycles of types (d) and (h). For cycles in this subset consider an additional parameter: $\xi_c^u = 1$, if there exists another cycle $c' \in \mathcal{C}$ involving only vertices of c (i.e. $V(c') \subseteq V(c)$) and that cycle c' was not attacked (i.e. $\zeta_{c'}^u = 1$). Using these parameters, the back-arcs recourse function can be rewritten as a linear function:

$$R_{\text{Back-arcs}}(x, u) = \sum_{c \in \mathcal{C} \setminus \bar{\mathcal{C}}} w_c x_c \zeta_c^u + \sum_{c \in \bar{\mathcal{C}}} x_c (\zeta_c^u + (w_c - 1)\xi_c^u) \tag{29}$$

When both $\zeta_c^u = \xi_c = 1$ there was no failure in cycle $c \in \overline{\mathcal{C}}$, and the objective coefficient is equal to the weight of the cycle, w_c. If $\zeta_c^u = 0$, but $\xi_c^u = 1$ then cycle c was attacked and partly recovered by back-arc recourse, so the value of this cycle in the objective function is $(w_c - 1)$.

Thus the robust exchange problem under back-arcs recourse is written as follows:

$$RE_{\text{Back-arcs}}(\mathcal{U}) := \max \quad Z \tag{30}$$

$$\text{s.t.} \quad Z \leq \sum_{c \in \mathcal{C} \setminus \bar{\mathcal{C}}} w_c \zeta_c^u x_c + \sum_{c \in \bar{\mathcal{C}}} (\zeta_c^u + (w_c - 1)\xi_c^u) x_c \quad \forall u \in \mathcal{U} \tag{31}$$

$$\sum_{c\in\mathcal{C}:v\in V(c)} x_c \leq 1 \qquad \forall v \in V$$
$$x \in \{0,1\}^{|\mathcal{C}|}$$

In the following section we will discuss how to solve the problem in Step 3 (the adversary's problem) of the delayed scenario generation algorithm.

4.3.1 Solving the adversary's problem

Recall that homogeneous failure is assumed, and let uncertainty budget $b = \lfloor p\%(|N| + |A|)\rfloor$. Similarly to simple recourse we transform ζ^u and ξ^u into decision variables ζ_c for $c \in \mathcal{C}$ and ξ_c for $c \in \bar{\mathcal{C}}$. Variable ξ_c is set to 1 if cycle or chain c was not attacked, or was attacked only once. The adversary's problem is rewritten as:

$A_{\text{Back-arcs}}(x) :=$

$$\min_{\zeta,\xi} \sum_{c\in\mathcal{C}\setminus\bar{\mathcal{C}}} w_c x_c \zeta_c + \sum_{c\in\overline{\mathcal{C}}} x_c(\zeta_c + (w_c - 1)\xi_c\ ) \tag{32}$$
$$\text{s.t.} \sum_{c\in\mathcal{C}} \zeta_c + \sum_{c\in\overline{\mathcal{C}}} \xi_c \geq |\mathcal{C}| + |\overline{\mathcal{C}}| - b \tag{33}$$
$$\zeta_c \leq \xi_c \qquad \forall c \in \overline{\mathcal{C}} \tag{34}$$
$$\xi_c \leq 1 \qquad \forall c \in \mathcal{C} \tag{35}$$
$$\zeta_c \in \{0,1\} \qquad \forall c \in \mathcal{C}$$
$$\xi_c \in \{0,1\} \qquad \forall c \in \overline{\mathcal{C}}$$

Here, the objective (32) is to minimize the number of patients receiving a kidney under the back-arcs recourse as specified by (29). Constraints (33) specify the adversary's uncertainty 'budget' in the homogeneous failure setting. Constraints (34) guarantee that chains and cycles of set $\bar{\mathcal{C}}$ are not canceled entirely unless they contain two failures. Constraints (35) are redundant but are added to aid the exposition of the remaining of our analysis.

Denote by $A'_{\text{Back-arcs}}(x)$ the relaxed adversary's problem with no binary requirements on variables ζ and ξ. We now provide greedy strategy to built a feasible solution to $A'_{\text{Back-arcs}}(x)$ and prove that such solution is optimal. As a consequence, a lower bound to $A_{\text{Back-arcs}}(x)$ is determined. We also compute a binary feasible solution to $A_{\text{Back-arcs}}(x)$ for which the objective value coincides with the determined lower bound. As a conclusion, a solution to $A_{\text{Back-arcs}}(x)$ is immediate after solving its relaxation.

Let us partition the set of cycles and chains that need two attacks to be completely eliminated for all their vertices into two subsets: $\bar{\mathcal{C}} = \bar{\mathcal{C}}^N \cup \bar{\mathcal{C}}^P$, where $\bar{\mathcal{C}}^N$ is a set of chains with configuration as in Figure 4 (d), and $\bar{\mathcal{C}}^P$ is a set of cycles with configuration as in Figure 4 (h).

A cycle or chain c in a solution, $x_c = 1$, can be characterized by the ratio $\rho = \dfrac{w_c}{b(c)}$ between w_c, which corresponds to the decrease in the objective function of $A'_{\text{Back-arcs}}(x)$ by completely eliminating c, and $b(c)$, the number of vertices that must fail in order to completely eliminate c. ρ can be interpreted as "gain" of adversary per "unit" of attack. By using this ratio, the cycles in a solution x can be partitioned into the following sets C_ρ, according to the value of ρ:

$$\begin{aligned}
C_3 &= \{c \in \mathcal{C} - \bar{\mathcal{C}}^P : w_c = 3 \text{ and } x_c = 1\}\\
C_2 &= \{c \in \mathcal{C} - \bar{\mathcal{C}}^N : w_c = 2 \text{ and } x_c = 1\}\\
C_{1.5} &= \{\bar{\mathcal{C}}^P : x_c = 1\}\\
C_1 &= \{c \in \mathcal{C} : w_c = 1 \text{ and } x_c = 1\}\\
C'_1 &= \{c \in \bar{\mathcal{C}}^N : x_c = 1\}.
\end{aligned}$$

In this way, a natural greedy strategy to built a solution to $A'_{\text{Back-arcs}}(x)$ is described in Theorem 1 (see the proof in Appendix 8.2).

Theorem 1 *Consider the setting with back-arcs recourse and homogeneous failure. Let $k = 3$, x be an arbitrary feasible solution to the robust exchange problem, and $b = \lfloor p(|N| + |A|) \rfloor$. An optimal solution to $A'_{Back\text{-}arcs}(x)$ is described below depending on the value of b.*

1. *If $b \leq |C_3|$, set $\zeta_c = 0$ for b cycles of C_3 and the remaining variables of $A'_{Back\text{-}arcs}(x)$ equal to 1.*
2. *If $|C_3| < b \leq |C_3| + |C_2|$, set $\zeta_c = 0$ for all $c \in C_3$, $\zeta_c = 0$ for $b - |C_3|$ chains and/or cycles of C_2 and the remaining variables of $A'_{Back\text{-}arcs}(x)$ equal to 1.*
3. *If $|C_3| + |C_2| < b \leq |C_3| + |C_2| + 2|C_{1.5}|$, set $\zeta_c = 0$ for all cycles of $C_3 \cup C_2$, $\zeta_c = \xi_c = 0$ for $\lfloor \frac{b - |C_3 \cup C_2|}{2} \rfloor$ cycles of $C_{1.5}$, $\zeta_c = \xi_c = \frac{1}{2}$ for one distinct cycle of $C_{1.5}$ if $b - |C_3 \cup C_2|$ is odd and the remaining variables of $A'_{Back\text{-}arcs}(x)$ equal to 1.*
4. *If $|C_3| + |C_2| + 2|C_{1.5}| < b \leq |C_3| + |C_2| + 2|C_{1.5}| + |C_1|$, set $\zeta_c = 0$ for all $c \in C_3 \cup C_2 \cup C_{1.5}$, $\xi_c = 0$ for all $c \in C_{1.5}$, $\zeta_c = 0$ for $b - |C_3 \cup C_2| - 2|C_{1.5}|$ elements of C_1, and the remaining variables of $A'_{Back\text{-}arcs}(x)$ equal to 1.*
5. *If $|C_3| + |C_2| + 2|C_{1.5}| + |C_1| < b \leq |C_3| + |C_2| + 2|C_{1.5}| + |C_1| + 2|C'_1|$, set $\zeta_c = 0$ for all $c \in C_3 \cup C_2 \cup C_{1.5} \cup C_1$, $\xi_c = 0$ for all $c \in C_{1.5}$, $\zeta_c = \xi_c = 0$ for $\lfloor \frac{b - |C_3 \cup C_2| - 2|C_{1.5}| - |C_1|}{2} \rfloor$ elements of C'_1, $\zeta_c = 0$ for distinct element of C'_1 if $b - |C_3 \cup C_2| - 2|C_{1.5}| - |C_1|$ is odd and the remaining variables of $A'_{Back\text{-}arcs}(x)$ equal to 1.*
6. *If $b > |C_3| + |C_2| + 2|C_{1.5}| + |C_1| + 2|C'_1|$, set $\zeta_c = 0$ for all $c \in C_3 \cup C_2 \cup C_{1.5} \cup C_1 \cup C'_1$, $\xi_c = 0$ for all $c \in C_{1.5} \cup C'_1$.*

Corollary 1 *In case of back-arcs recourse, homogeneous failure and $k = 3$, the optimal value of $A_{Back\text{-}arcs}(x)$ is equal to the optimal value of $A'_{Back\text{-}arcs}(x)$ rounded up.*

Proof. The optimal value for $A'_{\text{Back-arcs}}(x)$ rounded up gives a lower bound to the adversary's problem $A_{\text{Back-arcs}}(x)$ and can be obtained through Theorem 1. Note that case 3 of Thereom 1 is the only one for which the optimal solution of $A'_{\text{Back-arcs}}(x)$ might fail to be binary (and thus, unfeasible to $A_{\text{Back-arcs}}(x)$). In this way, it remains to show that under this situation, by rounding up the optimal value of $A'_{\text{Back-arcs}}(x)$, the optimal value for $A_{\text{Back-arcs}}(x)$ is obtained.

For case 3, at most two variables, ζ_c and ξ_c for some $c \in C_{1.5}$, are equal to $\frac{1}{2}$. By making $\zeta_c = 0$ and $\xi_c = 1$, the solution becomes a feasible solution to $A_{\text{Back-arcs}}(x)$ and the objective value increases by $\frac{1}{2}$ which is equal to the optimal value of $A'_{\text{Back-arcs}}(x)$ rounded up. □

When applying Algorithm 1 to solve the robust exchange problem with back-arc recourse, Step 3 implies the cancellation of cycles and chains according to the greedy strategy defined in Theorem 1. Similarly to simple recourse (see the end of Section 4.2), from the set of cycles and chains to be canceled we create a scenario that interdicts the vertices that participate in the largest number of cycles.

4.3.2 MIP for solving the back-arcs recourse robust exchange problem

Corollary 1 can be used to build a mixed integer program for solving the robust exchange problem with back-arcs recourse. Recall that $A'_{\text{Back-arcs}}(x)$ denotes the relaxed adversary's problem. By making this relaxation, the adversary has a larger set of feasible strategies and thus is potentially inducing more damage to the planned solution. This is the reason why we get a lower bound to the robust exchange problem by relaxing the integrality of the adversary's variables. By applying strong duality on $A'_{\text{Back-arcs}}(x)$ the following linear formulation can be build:

$$MIP'_{\text{Back-arcs}}(\mathcal{U}) :=$$

$$\max_{x,v} \big(|\mathcal{C}| + |\overline{\mathcal{C}}| - b\big)\, r - \sum_{c \in C(3)} h_c \tag{36}$$

$$
\begin{aligned}
\text{s.t.} \sum_{c\in\mathcal{C}:v\in V(c)} x_c &\leq 1 && \forall v \in V \\
r - h_c &\leq w_c x_c && \forall c \in \mathcal{C}\backslash\overline{\mathcal{C}} && (37)\\
r + \bar{h}_c - h_c &\leq (w_c - 1)x_c && \forall c \in \overline{\mathcal{C}} && (38)\\
r - \bar{h}_c &\leq x_c && \forall c \in \overline{\mathcal{C}} && (39)\\
x_c &\in \{0,1\} && \forall c \in \mathcal{C} \\
r &\geq 0 \\
h_c &\geq 0 && \forall c \in \mathcal{C} \\
\bar{h}_c &\geq 0 && \forall c \in \overline{\mathcal{C}}
\end{aligned}
$$

where r, $\bar{h}_c$ and h_c are the dual variables associated with constraints (33), (34) and (35), respectively.

The above formulation is a straightforward MIP that provides a lower bound to the original robust exchange problem. In fact, we will prove that from the optimal adversary's relaxed solution, a binary feasible interdiction can be determined such that the objective function does not increase by more than $\frac{1}{2}$, and thus, the optimal solution for the back-arcs recourse is attained. Before that, we illustrate this situation in the following example.

Example 3 *Consider the market graph represented in Figure 5 in which at most 2 nodes can fail. We have the set of cycles* $C(3) = \{c_1 = \langle 1,2\rangle, c_2 = \langle 2,3\rangle, c_3 = \langle 1,3\rangle, c_4 = \langle 4,5\rangle, c_5 = \langle 1,2,3\rangle, c_6 = \langle 3,2,1\rangle\}$ *and* $C''(3) = \{c_5, c_6\}$.

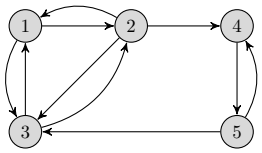


Figure 5: Compatibility graph of example 3

The optimal solution to the robust exchange problem with back-arcs recourse is to select cycles c_4 *and* c_5 *with objective value equal to 2. Although the optimal solution for the relaxed robust exchange problem* $MIP'_{Back\text{-}arcs}(\mathcal{U})$ *leads to the same planned solution, the objective value is equal to 1.5, since the adversary variables need not be binary and will be chosen as follows:* $\zeta_{c_1} = 1, \zeta_{c_2} = 1, \zeta_{c_3} = 1, \zeta_{c_4} = 0, \xi_{c_5} = \zeta_{c_5} = \frac{1}{2}, \xi_{c_6} = \zeta_{c_6} = 1$.

Corollary 2 *In case of back-arcs recourse, homogeneous failure and* $k = 3$*, the optimal value for the robust exchange problem* $RE_{Back\text{-}arcs}(\mathcal{U})$ *is equal to the optimal value of the relaxed robust exchange problem* $MIP'_{Back\text{-}arcs}(\mathcal{U})$ *rounded up.*

Proof. Recall that by Corollary 1 the optimal value of $A_{\text{Back-arcs}}(x)$ is equal to the optimal value of $A'_{\text{Back-arcs}}(x)$ rounded up. We prove that the result propagates to the objective values of $RE_{\text{Back-arcs}}(\mathcal{U})$ and $MIP'_{\text{Back-arcs}}(\mathcal{U})$. Let x^* and x^{**} be the optimal solutions to $RE_{\text{Back-arcs}}(\mathcal{U})$ and $MIP'_{\text{Back-arcs}}(\mathcal{U})$, respectively. By construction, it holds that

$$A'_{\text{Back-arcs}}(x^*) \leq A'_{\text{Back-arcs}}(x^{**}) \leq A_{\text{Back-arcs}}(x^{**}) \leq A_{\text{Back-arcs}}(x^*).$$

Since the value of $A_{\text{Back-arcs}}(x^{**})$ is integer, rounding up $A'_{\text{Back-arcs}}(x^*)$ we get a value less or equal to $A_{\text{Back-arcs}}(x^{**})$. This enables us to conclude that x^{**} must be optimal to $RE_{\text{Back-arcs}}(\mathcal{U})$. □

4.4 Full recourse

In this section discuss the application of Algorithm 1 for solving the robust exchange problem with full recourse. To start we consider a reformulation of the recurse function, that will allow us to reformulate the robust exchange problem $RE_{Full}(\mathcal{U})$ as a linear integer program.

The full recourse function $R_{Full}(x, u)$ is defined by problem (12)–(15). Let us consider the following additional decision variables:

$$y_v^u = \begin{cases} 1 & \text{if vertex } v \in V \text{ is selected in both the planned solution} \\ & \text{and the final solution under scenario } u \in \mathcal{U}, \\ 0 & \text{otherwise.} \end{cases}$$

Then we can rewrite the full recourse function as:

$$\begin{aligned} R_{\text{Full}}(x,u) := \max_{x^u, y^u} \quad & \sum_{v \in V} y_v^u && && (40)\\ \text{s.t.} \quad & y_v^u \le \sum_{c: v \in V(c)} x_c && \forall\ v \in V && (41)\\ & y_v^u \le \sum_{c: v \in V(c)} x_c^u \le 1 && \forall\ v \in V^u && (42)\\ & x_c^u \le \zeta_c^u && \forall c \in \mathcal{C} && (43)\\ & y^u \in \mathbb{R}_+^{|V|} \\ & x^u \in \{0,1\}^{|\mathcal{C}|} \end{aligned}$$

As before, the objective (40) is to maximize the number of vertices in the intersection of the planned and the final solution given the scenario $u \in \mathcal{U}$. Constraints (41) check whether a node is in the planned solution and constraints (42) check whether a node is in the final solution. Constraints (43) ensure that only available cycles and chains are used on the second stage.

The advantage of this formulation is that we can now again replace the recourse function in the robust exchange problem (4)-(6) and write it as the following *linear* problem:

$$\begin{aligned} RE_{\text{Full}}(\mathcal{U}) := \max \quad & Z \\ \text{s.t.} \quad & Z \le \sum_{v \in V} y_v^u && \forall\ u \in \mathcal{U} && (44)\\ & y_v^u \le \sum_{c: v \in V(c)} x_c \le 1 && \forall\ u \in \mathcal{U}, v \in V && (45)\\ & y_v^u \le \sum_{c: v \in V(c)} x_c^u \le 1 && \forall\ u \in \mathcal{U}, v \in V^u && (46)\\ & x_c^u \le \zeta_c^u && \forall c \in \mathcal{C}, u \in \mathcal{U} && (47)\\ & Z \in \mathbb{R}_+ \\ & y^u \in \mathbb{R}_+^{|V|} && \forall\ u \in \mathcal{U} && (48)\\ & x, x^u \in \{0,1\}^{|\mathcal{C}|} && \forall u \in \mathcal{U}. && (49) \end{aligned}$$

In the following section we present the approach for solving of adversary's problem for full recourse, when applying Algorithm 1. Notice, that differently from simple and back-arcs recourse, where adding a scenario u to the RE_{Simple} and $RE_{Back-arcs}$ formulations corresponds to generating a row, for the full recourse the addition of scenario implies simultaneously generating the relevant rows in (44), (45) and (46) and the relevant columns associated with the variables y^u and x^u.

4.4.1 Solving the adversary's problem

The adversary's problem for the full recourse reads as:

$$A_{\text{Full}}(x) := \min_{u \in \mathcal{U}} \quad R_{\text{Full}}(x, u) \tag{50}$$

It was proven in [28] that this problem is NP-hard. In what follows we develop a branch and bound method for solving exactly the adversary's problem $A_{\text{Full}}(x^*)$ (Algorithm 2).

Let x^* denote an optimal solution to the robust exchange problem $RE(\bar{\mathcal{U}})$. The algorithm does branching on the vertices and arcs in the exchange graph D. It starts by initializes the queue Q of active nodes of the search tree with root node t, which has no arcs and vertices fixed: $F^0(t) = F^1(t) = \emptyset$ (see lines 1-4).

The algorithm terminates, returning the best found solution, if the list Q of active nodes of the tree is empty (lines 5-7). Otherwise, it chooses node $t \in Q$ to explore (see line 8), in our implementation, by applying depth first search strategy. Scenario u^t is formed by vertices and arcs, interdicted on the path from root node to t (i.e., those belonging to set $F^0(t)$) and filled to have maximal attack (line 9). For the case of homogeneous failure, the latter implemented by interdicting vertices and arcs, that are not fixed yet (do not belong to $F^0(t)$ or $F^1(t)$) until uncertainty budget b is available (i.e. until $|V^u| + |A^u| = |V| + |A| - b$. To choose elements to be interdicted we applied simple greedy strategy, cancelling the cycles and chains with the maximum value. Objective value of recourse function $R_{\text{Full}}(x^*, u^t)$ provides an integral upper bound for node t, while the lower bound LB^t is obtained by objective value of adversary's problem for simple recourse $A_{Simple}(x, u^t)$ (see lines 9-12).

If the scenario u^t already leads to an objective function smaller than $RE(\bar{\mathcal{U}})$ (line 19), the algorithm stops as that is the goal of this scenario generation subproblem (recall Step 3 of Algorithm 1 and Proposition 2). Whenever for any node in the search tree the lower bound is no better than the best upper bound found so far or the uncertainty budget is fully explored, that node's subtree can be pruned (see lines 22–24).

Finally, if the node was not pruned, the branching is performed on the element τ (vertex or acr) from scenario u^t ($\tau \notin V^u \cup A^u$) that have not been fixed yet (do not belong to $F^0(t)$ or $F^1(t)$), and two branches are created (see lines 25-29).

There may be proposed different ways on improvement the bounds for the nodes in the tree. One of them is presented in Appendix 9 for the calculation of better upper bounds that can be particularly good for nodes near the root of the branch-and-bound tree.

5 Refinement of the robust solution

In KEP there may be multiple solutions that are optimal with respect to the chosen objective. Preliminary experiments suggest that this may also happen for our robust exchange models.

The most common approach to deal with multiple optimal solutions is to use a set of tie-breaking rules or secondary criteria [35, 27]. A set of multiple objectives may be combined into a single objective function by including a separate term for each criterion under consideration. Each term is then multiplied with the relative weight attached to the criterion it models. It is very common in kidney exchange for the criteria to be hierarchically ordered [18, 35, 32]. The objective weights should then be scaled such that the first criterion is indeed more important than the second, the second criterion more important than the third, etc. Alternatively, in case of hierarchical criteria, an iterative lexicographic approach may be considered [27].

In Section 6, we compare the deterministic problem with no vertices failure (only the best scenario $\mathcal{U} = \{N \cup A\}$ is taken into account) and worst case scenario for at most b vertices failure (takes into account all scenarios $\mathcal{U} := \{u \subseteq V \cup A : |V^u| \geq |V| - b, |A^u| = |A|\}$). Frequently, an optimal solution to the deterministic problem performs equally to an optimal robust solution in the worst case scenario. This

Algorithm 2 Solving the scenario generation subproblem $A_{\text{Full}}(x^*)$

Require: solution x^*
Ensure: Optimal solution (z^*, u^*) for the adversary's problem $A_{Full}(x^*)$

`//Initialization`
Create a queue Q with one node t,
$F^0(t) = \emptyset$ and $F^1(t) = \emptyset$ sets of vertices fixed to 0 and 1, respectively, defining branches of the tree.
Solution u^* = `None`
Objective value $z^* \leftarrow \infty$.
`//Termination`
if $Q = \emptyset$ **then**
  return (z^*, u^*)
end if
`//Problem selection and relaxation`
Select and remove a node t from Q
Construct scenario u^t: $u^t = F^0(t)$ and fill it to have maximal attack
if $|F^0(t)| \leq b$ **then**
  $UB^t \leftarrow R_{\text{Full}}(x^*, u^t)$
  $LB^t \leftarrow A_{Simple}(x^*, u^t)$
  if $UB^t \leq z^*$ **then**
    Update best solution: $u^* \leftarrow u^t$, $z^* \leftarrow UB^t$
  end if
else
  Go to line 5
end if
`//Optimality checking`
if $UB^t \leq RE(\bar{\mathcal{U}})$ **then**
  return (z^*, u^*)
end if
`//Pruning`
if $|F^0(t)| = b$ **or** $LB^t \geq z^*$ **then**
  Go to line 5
end if
`//Branching`
Choose an element τ (a vertex or an arc) from u^t and create two new nodes:
t^0, $F^0(t^0) \leftarrow F^0(t) \cup \tau$, $F^1(t^0) \leftarrow F^1(t)$
t^1, $F^0(t^0) \leftarrow F^0(t)$, $F^1(t^0) \leftarrow F^1(t) \cup \{\tau\}$
Add new nodes: $Q \leftarrow Q \cup \{t^0, t^1\}$
Go to line 5

traduces in the fact that there are multiple optimal robust solutions and thus we would like to select the one among them that performs best in the optimistic scenario.

In order to select the optimal solution for the robust exchange problem that performs best for the optimistic scenario we replace the objective function in (4) by

$$\max_{Z,x} \quad Z + \varepsilon \sum_{c \in \mathcal{C}} w_c x_c$$

with $\varepsilon > 0$.

The value of ε must be carefully chosen in order to avoid to get kidney exchange programs that are not robust, i.e, the second term in the new objective function should never be greater than the first term. Therefore, it is sufficient to set

$$\varepsilon = \frac{1}{\sum_{v \in V} \max_{c: v \in V(c)} \frac{w_c}{|c|}} \tag{51}$$

Note that we can also add the term $\varepsilon \sum_{c \in \mathcal{C}} w_c x_c$ to the objective function of the adversary's problem (19). As variables x are fixed in the adversary's problem, this is equivalent to adding a constant. In this way the results we obtained for solving the robust exchange problem for the various forms of recourse still apply.

In a similar way, we can use objective weights to assign priority to specific groups of agents, such as highly sensitized patients in kidney exchange. In case of the full recourse, we can also use a more efficient formulation than adding another term to the objective function. The structure of the recourse objective (40) allows to replace the unit objective weights of the y variables by scaled weights, such that the desired groups of agents are prioritized, e.g. highly sensitized patients. Let P^H denotes the set of pairs with highly sensitized patients. We replace the full recourse objective function (40) by:

$$R^H_{Full}(x,u) = \max_{x^u, y^u} \quad \sum_{v \in P^H} y^u_v + \sum_{v \in V \setminus P^H} \varepsilon y^u_v + \varepsilon^2 \sum_{c \in \mathcal{C}} w_c x_c \tag{52}$$

where ε is defined by (51).

As highly sensitized patients have a particularly high probability of match failure compared to non-highly sensitized patients (see [26]), we explicitly consider arc failure for this patients' group.

6 Computational results

In this section, we present computational experiment to validate the effectiveness of proposed robust models and compare solution approaches. In order to evaluate the impact of robustness, we compare the robust solutions for the different recourse policies to the deterministic ones. For all recourse policies the delayed scenario generation method was implemented for solving the robust exchange problem, as detailed in sections 4.2, 4.3 and 4.4. The problems for simple and back-arcs resources were also solved through their mixed integer programs reformulations(see sections 4.2.1 and 4.3.2).

We performed experiments for the particular case of the homogeneous failure, where the set of scenarios defined by at most b failing vertices $\mathcal{U} = \{u = V^u \cup A^u : |V^u| \geq |V| - b, |A^u| = |A|\}$ for $b = 1, \ldots, 4$.

6.1 Instance generator

Test instances for the computational experiments were generated by the simulator in [38] that uses US population data from the United Network for Organ Sharing (UNOS). The simulator generates patients with a random blood type, sex, and probability of being crossmatch incompatible (this probability is called the

PRA level) 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. Table 1 summarizes the parameters of the generation as described in [38]. As the original simulator did not include altruistic donors, we add to each pool a fixed percentage of altruistic donors (generated as above but without assignment to a patient).

Prob. blood type A	.3373
Prob. blood type B	.1428
Prob. blood type AB	.0385
Prob. blood type O	.4814
Prob. low PRA (5 %)	.7019
Prob. medium PRA (10 %)	.2
Prob. high PRA (90 %)	.0981
Prob. female	.409
Prob. spousal donor*	.4897
% altruistic donor**	4.5
* Applies to female patients only. Spousal PRA := 1 - .75 (1 - PRA)	
** Original simulator did not have altruistic donors	

Table 1: Parameters used for generation of instances by generator from [38]

For our experiments we generated 30 instances with 20, 50 and 100 vertices. Table 2 summarizes some characteristics of these instances: the number of nodes, $|V|$, the average number of arcs, $|A|$, and of cycles and chains, $\mathcal{C}$, also partitioned into sets of cycles and chains with 0, 1, 2 or 3 back-arcs: $\mathcal{C}_k^P$ and $\mathcal{C}_k^N$, respectively, where k is the number of back-arcs.

$\|V\|$	$\|A\|$	$\|\mathcal{C}\|$	$\mathcal{C}_0^P$	$\mathcal{C}_1^P$	$\mathcal{C}_2^P$	$\mathcal{C}_3^P$	$\mathcal{C}_0^N$	$\mathcal{C}_1^N$	$\mathcal{C}_2^N$
20	90.40	49.07	10.43	7.87	10.27	2.20	6.07	9.83	2.40
50	576.13	733.7	74.00	133.93	138.30	34.00	88.60	208.80	56.07
100	2367.20	6085.77	350.83	1047.3	1186.83	308.87	565.77	1994.37	631.80

Table 2: Average characteristics of the instances.

6.2 Evaluation of recourse policies under nodes failure

All the methods and models were implemented with C#.NET programming language and using CPLEX 12.8 for solving LPs and MIPs. The experiments were run on a computer equipped with a 3.6 GHz Intel Core i7 processor with 16 GB of RAM memory.

The results for tree recourse policies are presented in table 3. As mentioned above, they are also compared to performance of the deterministic problem with anticipation of failures (columns "Determ. Problem") were the following values are shown: Z^D is the average objective value for 30 instance of a given size; Z_b^D is the average objective value in the worst case scenario for a given optimal solution for maximum number of failing nodes b.

As to recourse policies the following results are provided: Z^* is the average objective value for the robust problem $RE(\mathcal{U})$; rl% is the percentage of instances in which the loss in transplants (i.e. the difference between the the objective value of the optimal recourse solution x^* in deterministic problem, $Z^D(x^*)$, and number of actual transplants in the second stage Z^*) is lower than the maximum possible loss for given instance and failure budget b. The latter is defined as a sum of values (w_c) of b the most valuable cycles or chains. T_{MIP} and T_{DSG} are average CPU times (in seconds) using mixed integer program (for simple and back-arc

recourse) and delayed scenario generation Algorithm 1 (for all recourse policies), respectively; Ns is the average nubmer of scenarios generated by the latter. The computational time for each instance was limited by 3600s., and we provide in parenthesis the number of instances (out of 30) which were not solved within this limit. The best bound of optimal solution was considered as Z^* for those instances. In addition, table 4 provides the average gaps for unsolved instances for each method and recourse policy.

First of all, we notice that despite the results shown in section 3.3, for all robust policies, all the instances and all the failure budgets, with single exception of an instance with $|V| = 50$, $b = 4$ and Back-arcs recourse, the optimal robust solution x^* was also optimal in deterministic problem, i.e. $Z^D(x^*) = Z^D$.

For all the recourse policies the optimal value in the worst case scenario was better that the deterministic problem with no care on uncertainty (compare columns Z^* and column Z^D_b). The optimal values Z^* obtained for simple and back-arcs recourse are very similar. They only differ for instances with 20 vertices for $b = 1, 2$, and for instances with $|V| = 50$ and $b = 4$. This indicates that in the worst case scenario the additional flexibility for recovering from failures provided by the back-arcs recourse is insufficient to reduce the losses compared to what could be gained by anticipating failures as by the simple recourse. The main reason is that, in almost all instances, it is not possible to select only cycles and chains with sufficient back-arcs to recover from a single node or arc failure. Therefore, in the worst case any of the selected 'non-robust' cycles and chains will be cancelled. In the case of a single failure ($b = 1$) the maximum possible loss can sometimes be restricted with those recourse policies (see the instances of size 20). However, for larger instance sizes (50 and 100) the loss cannot be avoided by anticipation of failure (see columns rl% for simple and back-arcs recourses). In the case of two failures, again the maximum loss can be restricted if the instance size is small enough (in the instances of size 20 and 50). Interestingly, in case the number of failures is higher, the percentage of instances in which the loss can be reduced increases substantially.

The full recourse clearly outperforms the other two policies by the quality of the solution provided in the worst case scenario. The better performance can be observed for instances with 50 vertices and bigger uncertainty budget $b = 2, 3, 4$, where in more than half of instances the maximal loss was reduced (see column rl% for Full recourse).

As to computational time, and comparing mixed integer programs opposed to delayed scenario generation for Simple and Back-arcs recources, there is no significant difference between the two solution approaches for smaller number of failures $b = 1, 2$. The MIP is slightly outperforming DSG by CPU time for simple recourse, $|V| = 100$ and $b = 2$, and by the quality of bounds obtained in case of unsolved instance (see table 4) for Back-arcs recourse and $b = 2$. However, for $b = 3, 4$ the delayed scenario generation algorithm clearly outperforms the mixed integer programs in particular for bigger instances with 100 vertices. If compare T_{MIP} and T_{DSG} for $b = 4$, 9 instances were not solved to optimality by MIP for both recourse policies opposed to 5 instances for DSG, and the computational time for the solved instances is significantly lower. Moreover, the average gaps between lower and upper bounds are also better for DSG algorithm than for MIP.

For the full recourse policy both the size of instance and the maximum number of failures b are the bottleneck for the developed algorithm. There 13 (out of 30) unsolved within time limit instances for $|V| = 100$ and $b = 3$, and the average time of solved instances dramatically increases when compare, for example, instances with 50 and 100 vertices, and the results for $b = 2$ and $b = 3$.

7 Conclusions

In this research we have considered the centralized clearing of barter exchange markets in which proposed transactions must be verified before they can proceed. Proposed transactions may fail to go forward if verification fails or if a participant withdraws. We have modeled the clearing problem in these markets as a vertex-disjoint cycle packing problem in an unreliable graph. The arcs and nodes of this graph are subject to failure.

Our research has many natural and interesting applications, of which kidney exchange is probably the most important. Deciding which donors get matched to which patients in kidney exchange can be a matter of life and death. Unfortunately, the current algorithms employed to clear kidney exchanges often leave

	Determ. problem		Simple recourse					Back-arcs recourse					Full recourse			
N	z^D	z_b^D	Z^*	rl%	T_{MIP}	T_{DSG}	Ns	Z^*	rl%	T_{MIP}	T_{DSG}	Ns	Z^*	rl%	T_{DSG}	Ns
								$b = 1$								
20	7.43	4.93	5.00	6.7	0.09	0.03	2.3	5.03	10.0	0.04	0.03	2.3	5.27	30.0	0.09	2.0
50	25.03	22.03	22.03	0.0	0.17	0.12	2.7	22.03	0.0	0.21	0.19	3.0	22.27	23.3	0.72	2.4
100	54.33	51.33	51.33	0.0	2.16	1.58	4.2	51.33	0.0	16.23	1.96	4.2	51.53	20.0	14.25	3.0
								$b = 2$								
20	7.43	2.97	3.10	13.3	0.09	0.04	2.6	3.17	16.7	0.06	0.02	2.5	3.40	33.3	0.15	2.1
50	25.03	19.07	19.20	13.3	0.85	0.39	4.6	19.20	13.3	1.18	0.44	5.7	19.63	50.0	4.36	2.5
100	54.33	48.33	48.37	3.3	(1) 9.49	(1) 17.61	10.0	48.37	3.3	(1) 13.61	(1) 11.51	8.9	48.83	36.7	(1) 254.21	3.1
								$b = 3$								
20	7.43	1.27	1.40	10.0	0.04	0.03	2.6	1.40	10.0	0.04	0.02	2.4	1.53	20.0	0.31	2.0
50	25.03	16.13	16.47	20.0	2.36	1.69	7.9	16.47	20.0	3.03	5.12	10.9	17.10	63.3	(1) 109.42	2.4
100	54.33	45.33	45.40	3.3	(1) 65.47	(2) 40.39	26.2	45.40	3.3	(2) 109.71	(2) 50.37	18.8	46.33	50.0	(13) 1228.71	2.4
								$b = 3$								
20	7.43	0.47	0.57	10.0	0.11	0.03	2.2	0.57	10.0	0.11	0.02	2.1	0.67	16.7	0.78	1.9
50	25.03	13.30	13.80	23.3	30.04	11.26	13.9	13.85	26.7	60.10	53.04	22.9	14.73	76.7	(3) 238.22	2.3
100	54.33	42.33	42.43	3.3	(9) 379.40	(3) 169.86	32.1	42.43	3.3	(9) 595.62	(5) 138.26	30.0	44.40	93.3	(26) 1247.60	2.1

Table 3: Average performance characteristics for tree recourse policies, compared to Deterministic problem before failure and after failure.

		Simple				Back-arcs				Full	
		MIP		DCG		MIP		DCG		DCG	
N	b	$N*$	$gap\%$	$N*$	$gap\%$	$N*$	$gap\%$	$N*$	$gap\%$	$N*$	$gap\%$
100	2	1	2.79	1	1.60	1	2.79	1	3.97	1	6.78
50	3	0		0	-	-	-	-	-	1	
100	3	1	6.81	2	2.39	2	5.30	2	3.17	13	3.61
50	4	0	-	0	-	-	-	-	-	3	16.11
100	4	9	6.79	3	4.52	9	5.99	5	3.80	26	4.62

Table 4: Gap for unsolved instances for different approaches; N* is the number of unsolved instances, $gap\%$ for MIP is integrality gap, for DCG as an upper bound we considered the current solution of the $RE(\bar{\mathcal{U}})$ or its bound, the lower bound is the last solution of adversary's problem or its lower bound.

highly-sensitized patients, which are hard to match, without a transplant. It has been the need to protect the rare transplant opportunities for these highly-sensitized patients that has motivated us in particular to consider the concept of a "robust exchange".

Other methodologies that aim to take market uncertainty into account, such as maximizing the expected number of transplants, typically disadvantage highly-sensitized patients as transactions involving these patients tend to have a high probability of failure. Under our "robust exchange" methodology we aim to protect transactions against a large set of possible scenarios for failure. Our methodology allows in particular to protect the transactions for highly-sensitized patients.

In addition to protecting against failure, we explicitly consider the option of flexible response to failures. We do this by allowing recourse actions. We have considered three recourse policies - simple recourse, back-arcs recourse, and full recourse - which can be easily implemented in practice. Our clearing algorithm selects an optimal planned solution taking the possibility of recourse into account. If actual failures occur, our algorithm selects the optimal recourse action.

We have provided results for settings in which the problem of determining the optimal recourse action can be solved efficiently. Moreover, we have shown that for these settings the problem of determining the worst case scenario (taking into account the possibility of recourse) can also be solved efficiently. These results apply to the simple recourse and back-arcs recourse, when trading cycles and chains are limited to three agents and failure is considered to be homogeneous. For other settings and for the full recourse, we have developed an advanced methodology for delayed scenario generation. In this methodology row and column generation are combined with a branch-and-bound algorithm.

We have tested our algorithms on various instances generated by the most commonly used kidney exchange simulator based on US population data. Our computational results show that instances of realistic size (the size of current kidney exchange pools), can be solved within run times that are acceptable to practice. More importantly, our results show that in a substantial number of instances, it is possible to actually protect patients against failures that prevent them to undergo a transplant. In this regard, our algorithms may offer a significant improvement over current practice.

There are several opportunities to expand the research presented in this paper. Direct extensions include extending the experiments to different types of uncertainty sets that reflect non-homogeneous failure or that tail-off as scenarios become more extreme (as per the concept of globalized robustness [6]). Another direction would be to combine our solution approach with delayed generation of trading cycles and chains [1, 27]. This would be particularly advantageous if the market size grows far beyond what it is today or if the bound on the trading cycles and chains becomes large.

There also remain general challenges to barter exchange markets that are important to mention. Dynamic market clearing - in which the market is not cleared by accumulating batches of agents and then maximizing the transactions per batch as is done in present exchanges, but in which the market is cleared while taking future arrivals into account - is a problem that has received attention but has not yet been solved optimally. Our model of market uncertainty can ostensibly be extended to take future arrivals into account. Another

challenge is the internationalization of markets and the conflicts of interest that may arise between market participants (e.g. participation and incentive compatibility for hospitals and networks of hospitals in kidney exchange). Finally, we would like to mention that generalizations of our work could consider allowing monetary transfers and private information regarding agent preferences. While these factors may be less important in kidney exchange markets, they may be important in other markets such as house trading.

Acknowledgments: This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project "mKEP - Models and optimisation algorithms for multicountry kidney exchange programs" (POCI-01-0145-FEDER-016677), by FCT project SFRH/BPD/101134/2014. The second author thanks the support of Institute for data valorisation (IVADO).

8 APPENDIX

8.1 Proof of totally unimodality of the constraints for simple recourse of adversary's problem

In case of simple recourse and homogeneous failure, the constraint matrix associated to the adversary's problem $A_{\text{Simple}}(x)$ is totally unimodular.

In case of simple recourse and homogeneous failure, and after relaxing the integrality requirement (27) on the ζ variables, the constraint matrix associated with the adversary's problem as specified by (25)-(26) in standard form is of the form

$$\begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ -1 & 0 & 0 & \dots & 0 \\ 0 & -1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & -1 \end{pmatrix}. \tag{53}$$

Theorem 13.3 from [40], gives sufficient conditions to prove that matrix (53) is totally unimodular. The first requirement is that all the entries in the matrix belong to $\{-1, 0, 1\}$ which holds. Second, no more than two entries in the same column may be non-zero. This condition is also satisfied since each decision variable ζ_c appears exactly in two constraints: once in (25) and another when stating the bound $\zeta_c \leq 1$. Finally, it must be possible to partition the rows of the matrix into two sets A and B such that: *i)* if a column has two entries of the same sign, their rows are in different sets; *ii)* if a column has two entries of different signs, their rows are in the same set. This is easily fulfilled if A contains all the rows and B is the empty set.

8.2 Proof of Theorem 1

Note that for any $c \in C_3$ not selected in x, this is, $x_c = 0$, the adversary's variables can simply be set to be 1 without affecting its objective function. Thus, without loss of generality and for sake of simplicity, let C_3 be restricted to the elements selected in the planned solution x.

The feasibility of the solution described by the theorem to $A'_{\text{Back-arcs}}(x)$ can be directly checked.

It remains to prove optimality for this greedy solution. To that end, strong duality wil be used. We will built a solution for the dual of $A'_{\text{Back-arcs}}(x)$ such that the associated dual objective value coincides with the one of $A'_{\text{Back-arcs}}(x)$ under theorem solution, establishing optimality.

The dual of $A'_{\text{Back-arcs}}(x)$ is

$$\max_{v} \quad (|C_3| + |C_2| + 2|C_{1.5}| + |C_1| + 2|C'_1| - b)\, v_0$$

$$
\begin{aligned}
& -\sum_{c\in C_3} v_c - \sum_{c\in C_2} v_c - \sum_{c\in C_{1.5}} v_c - \sum_{c\in C_1} v_c - \sum_{c\in C_1'} v_c && && (54)\\
\text{s.t.}\quad & v_0 - v_c \le 3 && \forall c \in C_3\\
& v_0 - v_c \le 2 && \forall c \in C_2\\
& v_0 - v_c \le 1 && \forall c \in C_1\\
& v_0 + v_c^1 - v_c \le 2 && \forall c \in C_{1.5}\\
& v_0 + v_c^1 - v_c \le 1 && \forall c \in C_1'\\
& v_0 - v_c^1 \le 1 && \forall c \in C_1'\\
& v_0 - v_c^1 \le 1 && \forall c \in C_{1.5}\\
& v_0 \ge 0 &&\\
& v_c \ge 0 && \forall c \in C_3 \cup C_2 \cup C_{1.5} \cup C_1 \cup C_1'\\
& v_c^1 \ge 0 && \forall c \in C_{1.5} \cup C_1'
\end{aligned}
$$

As done the the theorem statement, we divide the proof in cases, according with the value of b.

1. In the described solution, the objective value of $A'_{\text{Back-arcs}}(x)$ is
$$3|C_3| - 3b + 2|C_2| + 3|C_{1.5}| + |C_1| + 2|C_1'|. \tag{55}$$
The value of (54) under the following dual feasible solution
$$
\begin{array}{llllllll}
v_0 = 3 & & & & & & & \\
v_c = 0 & \forall c \in C_3, & v_c = 1 & \forall c \in C_2, & v_c = 2 & \forall c \in C_1, & v_c{=}3 & \forall c \in C_{1.5}\\
v_c^1 = 2 & \forall c \in C_{1.5}, & v_c = 4 & \forall c \in C_1', & v_c = 2 & \forall c \in C_1' & &
\end{array}
$$
coincides with the primal value (55).
2. In the described solution, the objective value of $A'_{\text{Back-arcs}}(x)$ is
$$2|C_3| + 2|C_2| - 2b + 3|C_{1.5}| + |C_1| + 2|C_1'|. \tag{56}$$
The value of (54) under the following dual feasible solution
$$
\begin{array}{llllllll}
v_0 = 2 & & & & & & & \\
v_c = 0 & \forall c \in C_3, & v_c = 0 & \forall c \in C_2, & v_c = 1 & \forall c \in C_1, & v_c{=}1 & \forall c \in C_{1.5}\\
v_c^1 = 1 & \forall c \in C_{1.5}, & v_c = 2 & \forall c \in C_1', & v_c = 1 & \forall c \in C_1' & &
\end{array}
$$
coincides with the primal value (56).
3. In the described solution, the objective value of $A'_{\text{Back-arcs}}(x)$ is
$$\frac{3}{2}|C_3| + \frac{3}{2}|C_2| + 3|C_{1.5}| - \frac{3}{2}b + |C_1| + 2|C_1'|. \tag{57}$$
The value of (54) under the following dual feasible solution
$$
\begin{array}{llllllll}
v_0 = \frac{3}{2} & & & & & & & \\
v_c = 0 & \forall c \in C_3, & v_c = 0 & \forall c \in C_2, & v_c = \frac{1}{2} & \forall c \in C_1, & v_c{=}0 & \forall c \in C_{1.5}\\
v_c^1 = \frac{1}{2} & \forall c \in C_{1.5}, & v_c = 1 & \forall c \in C_1', & v_c = \frac{1}{2} & \forall c \in C_1' & &
\end{array}
$$
coincides with the primal value (57).
4. In the described solution, the objective value of $A'_{\text{Back-arcs}}(x)$ is
$$|C_3| + |C_2| + 2|C_{1.5}| + |C_1| - b + 2|C_1'|. \tag{58}$$
The value of (54) coincides with the primal value (58), if the dual (feasible) solution with $v_c = 1$ and the remaining dual variables equal to zero is considered.
5. This case is completely analogous to 4.
6. In the described solution, the objective value of $A'_{\text{Back-arcs}}(x)$ is zero. All dual variables equal to zero is a dual feasible solution with objective value equal to zero.

9 Refining the procedure to obtain better upper bounds

For some nodes in the branch-and-bound tree it is possible to obtain substantially better upper bounds than those obtained by solving $R_{\text{Full}}(x^*, u^t)$ (Step 3). This is particularly the case for nodes near the root of the branch-and-bound tree. In order to explain how to achieve these improved bounds, note that if the integrality of the recourse variables x^u in the full recourse problem $R_{\text{Full}}(x^*, \zeta)$ described by (12), (13), (14) is relaxed, the adversary's problem $A_{\text{Full}}(x^*)$ can be solved by rewriting the nonlinear min-max objective as a minimization problem by using the dual of the recourse problem. The resulting problem is a minimum vertex and arc cover problem with variable cost:

$$
\begin{aligned}
A'_{\text{Full}}(x) := \quad & \min_{\zeta, W} && \sum_{v \in V} \zeta_n W_n + \sum_{a \in A} \zeta_a W_a && && (59)\\
& \text{s.t.} && \sum_{n \in c} W_n + \sum_{a \in c} W_a \geq \sum_{n \in c} \sum_{c' \in \mathcal{C}: n \in c'} x_{c'} && \forall\ c \in \mathcal{C} && (60)\\
& && B\zeta \leq b && &&\\
& && \zeta \in \{0,1\}^{|N|} && &&\\
& && W \in \mathbb{R}_+^{|N|+|A|} && &&
\end{aligned}
$$

Here, the variables W are the duals of constraints (13) and (14). The objective (59) is to find a minimum cost cover. Constraints (60) imply that all cycles that include nodes selected in the first stage must be covered.

The nonlinear terms $\sum_{v \in V} \zeta_n W_n$ and $\sum_{a \in A} \zeta_a W_a$ in the objective (59) may be linearized by introducing a variable $T_n := \zeta_n W_n$ for each $v \in V$ and a variable $S_a := \zeta_a W_a$ for each $a \in A$, and by imposing the additional constraints $T_n \geq W_n - M(1 - \zeta_n)$ for all $v \in V$ and $S_a \geq W_a - M(1 - \zeta_a)$ for all $a \in A$, where M is some sufficiently large number. In this case, setting $M := k$ is sufficient because constraints (60) imply that neither any W_n nor any W_a ever need to be larger than k. Applying these adjustments, we obtain the following mixed integer program:

$$
\begin{aligned}
A'_{\text{Full}}(x) := \quad & \min_{\zeta, W, S, T} && \sum_{v \in V} T_n + \sum_{a \in A} S_a && &&\\
& \text{s.t.} && \sum_{n \in c} W_n + \sum_{a \in c} W_a \geq \sum_{n \in c} \sum_{c' \in \mathcal{C}: n \in c'} x_{c'} && \forall\ c \in \mathcal{C} &&\\
& && W_n + k\zeta_n - T_n \leq k && \forall\ v \in V &&\\
& && W_a + k\zeta_a - S_a \leq k && \forall\ a \in A &&\\
& && B\zeta \leq b && &&\\
& && \zeta \in \{0,1\}^{|N|} && &&\\
& && W \in \mathbb{R}_+^{|N|+|A|} && &&\\
& && S \in \mathbb{R}_+^{|A|} && &&\\
& && T \in \mathbb{R}_+^{|N|} && &&
\end{aligned}
$$

The difference between the bound obtained by solving $R_{\text{Full}}(x^*, u^t)$ and the bound obtained by solving $A'_{\text{Full}}(x^*)$ is that the former accurately takes into account the recourse actions but underestimates the adversary's potential by using the scenario u^t, whereas the latter overestimates the recourse actions because the recourse variables are relaxed but accurately takes into the adversary's potential to damage the planned solution.

Finally, note that it is possible to use the LP relaxations $R'_{\text{Full}}(x^*, u^t)$, $A'_{\text{Simple}}(x^*)$, and $A'_{\text{Back-arcs}}(x^*)$ when determining the bounds in the branch-and-bound procedure. While this may provide bounds that are less tight, it may save computation time. We have the following relationships:

$$
R'_{\text{Full}}(x^*, u^t) \geq R_{\text{Full}}(x^*, u^t),
$$

$$A'_{\text{Simple}}(x^*) \leq A_{\text{Simple}}(x^*),$$

$$A'_{\text{Back-arcs}}(x^*) \leq A_{\text{Back-arcs}}(x^*)$$

that hold in any branch-and-bound node t.

References

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

[2] David L. Alderson, Gerald G. Brown, W. Matthew Carlyle, and R. Kevin Wood. Solving Defender-Attacker-Defender Models for Infrastructure Defense. In R. Kevin Wood and Robert F. Dell, editors, *Proceedings of the 12th INFORMS Computing Society Conference: Research, Computing, and Homeland Defense*, pages 28–49. INFORMS, 2011.

[3] I. Ashlagi, P. Jaillet, and V. H. Manshadi. Kidney exchange in dynamic sparse heterogenous pools. *Working paper*, 2013.

[4] Alper Atamtürk and Muhong Zhang. Two-stage robust network flow and design under demand uncertainty. *Operations Research*, 55(4):662–673, 2007.

[5] P. Awasthi and T. Sandholm. Online stochastic optimization in the large: Application to kidney exchange. *Preprint*, 2009.

[6] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

[7] Aharon Ben-Tal, Alexander Goryashko, Elana Guslitzer, and Arkadi Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, 99(2):351–376, 2004.

[8] Aharon Ben-Tal and Arkadi Nemirovski. Robust truss topology design via semidefinite programming. *SIAM Journal on Optimization*, 7(4):991–1016, 1997.

[9] Aharon Ben-Tal and Arkadi Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[10] C. Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences*, 43(9):842–844, 1957.

[11] Dimitris Bertsimas and Constantine Caramanis. Finite adaptability in multistage linear optimization. *Automatic Control, IEEE Transactions on*, 55(12):2751–2766, 2010.

[12] Dimitris Bertsimas and Aurélie Thiele. Robust and data-driven optimization: Modern decision-making under uncertainty. *INFORMS Tutorials in Operations Research: Models, Methods, and Applications for Innovative Decision Making*, 2006.

[13] Michael S. Casey and Suvrajeet Sen. The scenario generation algorithm for multistage stochastic linear programming. *Mathematics of Operations Research*, 30(3):615–631, August 2005.

[14] Xin Chen, Melvyn Sim, and Peng Sun. A robust optimization perspective on stochastic programming. *Operations Research*, 55(6):1058–1071, 2007.

[15] Yanhua Chen, Yijiang Li, John D. Kalbfleisch, Yan Zhou, Alan Leichtman, and P. X. Song. Graph-based optimization algorithm and software on kidney exchanges. *Biomedical Engineering, IEEE Transactions on*, 59(7):1985–1991, 2012.

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

[17] Marry de Klerk, Karin M Keizer, Frans H J Claas, Marian Witvliet, Bernadette J J M Haase-Kromwijk, and Willem Weimar. The dutch national living donor kidney exchange program. *American journal of transplantation: official journal of the American Society of Transplantation and the American Society of Transplant Surgeons*, 5(9):2302–2305, September 2005.

[18] Marry De Klerk, Wilfred M Van Der Deijl, Marian D Witvliet, Bernadette J J M Haase-Kromwijk, Frans H J Claas, and Willem Weimar. The optimal chain length for kidney paired exchanges: an analysis of the dutch program. *Transplant international: official journal of the European Society for Organ Transplantation*, 23(11):1120–1125, November 2010.

[19] F. Delmonico, 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, and R. Rohrer. Donor kidney exchanges. *Am J Transplant*, (4):1628–1634, 2004.

[20] J Dickerson, A Procaccia, and T Sandholm. Dynamic matching via weighted myopia with application to kidney exchange. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. 2012.

[21] J Dickerson, A Procaccia, and T Sandholm. Failure aware kidney exchange. In *Proceedings of the ACM Conference on Electronic Commerce (EC)*. 2013.

[22] J. P. Dickerson, D. F. Manlove, B. Plaut, T. Sandholm, , and J. Trimble. Position-indexed formulations for kidney exchange. In *In: 17th ACM Conference on Economics and Computation, Maastricht, The Netherlands, 24-28 Jul.* ACM, 2016.

[23] John P. Dickerson, Ariel D. Procaccia, and Tuomas Sandholm. Price of fairness in kidney exchange. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1013–1020. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[24] Laurent El Ghaoui and Hervé Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18(4):1035–1064, 1997.

[25] Laurent El Ghaoui, Francois Oustry, and Hervé Lebret. Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization*, 9(1):33–52, 1998.

[26] K. M. Glorie, A. P. M. Wagelmans, and JJ van de Klundert. Iterative branch-and-price for large multi-criteria kidney exchange. *Econometric Institute report*, (2012-11), 2012.

[27] K. M. Glorie, A. P. M. Wagelmans, and JJ van de Klundert. Kidney exchange with long chains: an efficient pricing algorithm for clearing barter exchanges with branch-and-price. *Manufacturing & Service Operations Management*, (Forthcoming), 2014.

[28] K.M. Glorie, M. Carvalho, P. Bouman, A. Viana, and M. Constantino. *Clearing Barter Exchange Markets: Kidney Exchange and Beyond, Chapter VI.* PhD thesis, Erasmus University Rotterdam, 2014.

[29] Kristiaan M Glorie, Marry de Klerk, Albert P M Wagelmans, Joris J van de Klundert, Willij C Zuidema, Frans H J Claas, and Willem Weimar. Coordinating unspecified living kidney donation and transplantation across the blood-type barrier in kidney exchange. *Transplantation*, August 2013.

[30] Gagan Goel and Pushkar Tripathi. Matching with our eyes closed. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 718–727. IEEE, 2012.

[31] P. Hosteins, A. Viana, M. Constantino, X. Klimentova, and J.P. Pedroso. A bi-objective stochastic approach to kidney exchange programs with conditional value at risk. *Submitted to Omega: The International Journal of Management Science.*

[32] Beom Seok Kim, Yu Seun Kim, Soon Il Kim, Myoung Soo Kim, Ho Yung Lee, Yong-Lim Kim, Chan Duck Kim, Chul Woo Yang, Bum Soon Choi, Duck Jong Han, Yon Su Kim, Sung Joo Kim, Ha-Young Oh, and Dae Joong Kim. Outcome of multipair donor kidney exchange by a web-based algorithm. *Journal of the American Society of Nephrology*, 18(3):1000–1006, March 2007.

[33] X. Klimentova, J.P. Pedroso, and A. Viana. Maximising expectation of the number of transplants in kidney exchange programmes. *Computers & Operations Research*, 73:1–11, 2016.

[34] Panos Kouvelis and Gang Yu. *Robust discrete optimization and its applications*, volume 14. Springer, 1997.

[35] D. Manlove and G. O'Malley. 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.*, pages 271–282, 2012.

[36] Marco Molinaro and R. Ravi. *Kidney exchanges and the query-commit problem.* Manuscript, 2013.

[37] João Pedro Pedroso and Shiro Ikeda. Maximum-expectation matching under recourse. *CoRR*, abs/1605.08616, 2016.

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

[39] Allen L. Soyster. Technical note-convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations research*, 21(5):1154–1157, 1973.

[40] Kenneth Steiglitz and Christos H. Papadimitriou. Combinatorial optimization: Algorithms and complexity. *Printice-Hall, New Jersey*, 1982.