Recourse in Kidney Exchange Programs*

B. Smeulders[†], V. Bartier[‡], Y. Crama[§], F.C.R. Spieksma[¶]

August 2, 2019

Abstract

The problem to decide which patient-donor pairs in a kidney exchange program should undergo a *crossmatch* test is modelled as a two-stage stochastic optimization problem. We give an integer programming formulation of this so-called selection problem, and describe a solution method based on Benders decomposition. We extensively test various solution methods, and observe that the solutions, when compared to solutions found by recourse models, lead to an improvement in the expected number of transplants. We also investigate the computational efficiency of our approach as a function of different parameters, such as maximum cycle length and the presence of altruistic donors.

1 Introduction: kidney exchange programs

Mathematical optimization techniques have established themselves as an important and indispensable tool in guiding decisions in kidney exchange programs. There is a large and fast-growing amount of literature documenting various successful implementations of algorithms that find cycles and chains in appropriately defined *compatibility* graphs. The increased performance of such algorithms has led to a better usage of available human kidneys, and as a result, many lives have been positively impacted.

This paper focuses on the issue of dealing with incompatibilities that may reveal themselves *after* an intended transplant has been identified. This is an important issue; for example Dickerson et al. [2018] report that 93% of proposed matches fail in the UNOS program, for a wide variety of reasons. In the NHS Living Kidney Sharing Scheme, 30% of identified matches did not proceed to transplant between 2013-2017 [NHS, 2017a]. Here, we analyze how this phenomenon can be taken into consideration in optimization models. We propose a new, generic, integer programming formulation to identify the maximum expected number of transplants, and we perform extensive computational experiments with this model. The resulting outcomes give insights on how kidney exchange programs can best prepare for the challenges that arise when confronted with a posteriori incompatibilities.

In order to set the stage for our contribution, we first give a stylized description of the operation of a kidney exchange program; in this description we momentarily ignore many of the practical features that exist in real-life kidney exchange programs.

The preferred treatment for a patient with end-stage renal disease is receiving a kidney from a living human donor. Many patients have a donor, often a friend or family member, who has volunteered to donate one of their kidneys for a transplant. However, a donor must be *compatible*

*A very preliminary version of this work was presented at the MATCH-UP 2019 conference and has benefited from the comments of anonymous reviewers. In addition, discussions with participants of meetings induced by the COST-action CA15210 European Network on Collaboration for Kidney Exchange Programmes have usefully contributed to the paper.

[†]HEC Management School, University of Liège, Belgium; post-doctoral fellow of the F.R.S.-FNRS; email: bart.smeulders@uliege.be.

[‡]G-SCOP, Grenoble INP, France.

[§]HEC Management School, University of Liège, Belgium.

[¶]Department of Mathematics and Computer Science, Eindhoven University of Technology, the Netherlands.

with a patient for a transplant to be possible. Determining whether or not a donor is compatible with their corresponding patient is done by a preliminary screening, based on blood type and immunological properties. When the donor and the patient are not compatible, they may together decide to enter a kidney exchange program where more transplant opportunities become available by relying on the diversity of a larger pool of individuals. We refer to Gentry et al. [2011] for more information on this process. Thus, a kidney exchange program consists of a set of patient-donor pairs (the *pool*), for which compatibilities have been derived from the preliminary screening tests. The objective is then to identify sequences of potential kidney donations among pairs of the pool, whereby each patient in the sequence receives a kidney from the donor of the previous pair, while the associated donor donates a kidney to the next patient. This description implies that the sequence of donations must necessarily be cyclic. Exchange programs, however, may also allow for the presence of *non-directed donors* who are not associated with any patient. They may increase the range of feasible sequences by acting as starting points of sequences of donations which end with a patient receiving a kidney, while the associated donor is not involved in any transplant (and may subsequently act as a non-directed donor).

A natural and well-established way of describing the operation of a kidney exchange program is by considering a so-called *compatibility graph*. In this directed graph, a vertex is associated with each patient-donor pair and each non-directed donor. There is an arc from vertex i to vertex j if the donor associated with vertex i is compatible with the patient of pair j (according to preliminary screening). A k-cycle, that is, a directed cycle of length k in this graph indicates a sequence of k transplants that can be simultaneously performed, based on the compatibilities identified in the preliminary screening phase. Similarly, a k-chain, that is, a directed chain of length k originating at a vertex associated with a non-directed donor, also indicates a feasible sequence of k transplants. Typically, for logistical reasons, an upper bound is given on the length of the cycles and chains that can be considered for transplants. The problem faced by the kidney exchange program is then to find a collection of vertex-disjoint cycles and chains of bounded length, covering as many arcs as possible, and thus allowing for the largest possible number of transplants in the current pool (see Section 2.2 for a more formal definition). We will refer to this optimization problem as the *kidney exchange problem*, or KEP for short.

Clearly, the description above is a very stylized sketch of the operation of a kidney exchange program. In practice, a program may present many other features, for example, arcs may be weighted to prioritize certain types of patients or transplants. We refer to Gentry et al. [2011] and Anderson et al. [2015] for a more elaborate discussion, and to Biró et al. [2019] for an overview of kidney exchange programs in Europe.

The key issue that we address here is that, in kidney exchange programs, compatibilities arising from preliminary screening are rarely certain and must be assessed by further tests. Indeed, after solving the kidney exchange problem, that is, after having identified a set of cycles and chains intended to give rise to a number of transplants, it may turn out that, for various reasons, some of the transplants cannot take place. This may be the case because a patient has already received a kidney from another program, or is too ill to undergo surgery. Another frequent reason is that further compatibility tests, called *crossmatch tests*, which are carried out after the potential matches have been identified, reveal previously undetected incompatibilities. Such crossmatch tests must always be applied before any transplant is allowed to be performed. It is important to understand that because the crossmatch tests must evaluate specific characteristics of the donor and patient involved, they are complex, time consuming, and expensive. Therefore, these tests are currently only carried out once an intended transplant has been identified.

There is a significant probability that a crossmatch test detects an incompatible transplant (see Dickerson et al. [2016]). And of course, when this happens, it implies that not only this particular transplant cannot be carried out, but also the other transplants in the same cycle, or further in the chain, fail to be implemented.

Our main contribution in this paper is to explicitly identify the problem of selecting the set of arcs that should undergo the crossmatch tests, so as to maximize the expected number of transplants. We formulate it as a two-stage stochastic optimization problem. In the first stage, we select a set of arcs that each will undergo crossmatch tests. The problem of identifying this

set of arcs is called the *selection problem*, see Section 3. In the second stage, we simply solve the (deterministic) kidney exchange problem on the graph induced by the arcs that passed the crossmatch tests.

We summarize our contributions as follows.

- We formalize the selection problem, and argue that solving this problem is a key ingredient in obtaining solutions that maximize the expected number of transplants (Section 3).
- We show that the selection problem is NP-hard, even when the maximum allowed cycle length is equal to 2 (Section 4).
- We show how to apply Benders decomposition to a generic integer programming formulation of the selection problem (Section 5).
- We perform extensive computational experiments showing the impact of various modeling and algorithmic choices, and we compare our results with those obtained by different approaches (Section 6).

2 Problem statement

We provide a brief overview of earlier work on kidney exchange problems in Section 2.1. Next, we set the stage by giving a generic formulation of the kidney exchange problem in Section 2.2, and by detailing the stochastic version of the kidney exchange problem in Section 2.3.

2.1 Literature review

Seminal work on modelling kidney exchange programs through integer programming was initiated by Roth et al. [2004], Roth et al. [2006], Montgomery et al. [2006]. The deterministic kidney exchange problem is a difficult combinatorial optimization problem: as observed in Abraham et al. [2007], it is NP-hard when restricted to any fixed cycle length greater than 2. (For cycle length equal to 2, KEP can be solved in polynomial time as a maximum matching problem.) In practice, however, optimal solutions of appropriate integer programming formulations can be computed in acceptable running times for medium to large instances; see, e.g., Dickerson et al. [2016], Mak-Hau [2017] and Manlove and Omalley [2015] for recent references.

In order to take crossmatch tests and uncertainty into account, two broad classes of approaches have been proposed, namely, *adaptive* and *non-adaptive* approaches. In non-adaptive approaches, a subset of potential transplants is first selected and crossmatch tests are subsequently performed in parallel on all the arcs of these subsets. The arcs that pass the crossmatch test can finally be used to identify the transplants to be executed by the kidney exchange. Adaptive approaches allow for more rounds of tests. After each round of crossmatch tests, additional arcs are selected for testing and this choice is dependent on the successes and failures in the previous rounds. Eventually, the crossmatching phase terminates and the successful arcs are used to identify the transplants to be performed.

Non-adaptive approaches have received most of the attention in the literature and have led to various formulations of the stochastic problem. Dickerson et al. [2013, 2018] propose re-weighting cycles and chains to reflect the expected number of transplants that can be performed using those subgraphs, and they solve the associated weighted packing problem. A compact formulation of this packing problem is given by Dickerson et al. [2016]. Pedroso [2014] also re-weights cycles, but his model accounts for all arcs induced by each cycle, thus allowing for limited recourse at the cost of additional tests; Alvelos et al. [2015] provide a compact formulation of this packing model. Klimentova et al. [2016] propose another recourse scheme where overlapping cycles can be tested. We give a more explicit description of these recourse schemes in Section 2.3.

Another stream of literature focuses on approximation algorithms. Blum et al. [2013] develop such an algorithm for the case of 2-cycles where each patient is involved in at most two crossmatch tests; they prove that their algorithm finds near-optimal solutions in sufficiently large kidney

exchange pools. Blum et al. [2015] provide non-adaptive approximation algorithms requiring a constant number of tests per vertex and delivering a solution with expected value within a factor $\frac{(2/K)^2}{2/K+1}(1-\epsilon)$ of the expected optimal value for the case of K-cycles or K-chains. Assadi et al. [2016], Behnezhad et al. [2018] and other authors subsequently strengthened the result for 2-cycles.

Adaptive approximation algorithms for bounded cycles and chains have been introduced by Blum et al. [2015]. For cycles of length 2, the authors proposed an algorithm which returns a matching with expected value at most $(1-\epsilon)$ the expected optimal matching value after a constant number of rounds and a constant number of queries (i.e., crossmatch tests) per vertex. They further extended these results to the K-set packing problem for which they obtained a $\frac{2}{K}(1-\epsilon)$-approximation algorithm. Assadi et al. [2016] subsequently improved the results for 2-cycles.

In real-world practice, both adaptive and non-adaptive policies are actually used. The NHS in the United Kingdom computes solutions that can be adjusted if planned transplants do not pass the crossmatch test, in the spirit of the non-adaptive procedure of Pedroso [2014]. Specifically, the NHS prefers (ceteris paribus) 3-cycles with embedded 2-cycles over 3-cycles without embedded 2-cycles. In this way, if one of the transplants in the 3-cycle turns out to be infeasible, the 2-cycle can be performed instead [NHS, 2017b]. Smaller programs, such as the Dutch and Czech program, iterate between solving a KEP and crossmatching all transplants in the solution. This adaptive process terminates when all tests are successful and the number of transplants is thus maximized [Biró et al., 2019].

2.2 The deterministic kidney exchange problem

Let us now turn to a more formal definition of the deterministic kidney exchange problem. An instance of the problem is defined by a simple, directed graph $G = (V, A)$, and by two integers K and L. Each vertex in V represents either a patient-donor pair or a non-directed donor. An arc $(i, j) \in A$ represents a possible transplant of a kidney from the donor associated with vertex i to the patient associated with vertex j. We use $V(G')$ $(A(G'))$ to denote the set of all vertices (arcs) in a subgraph G'. Let C be the set of all directed cycles c in G with length $w_c = |V(c)| = |A(c)|$ at most K. Similarly, let H be the set of all chains h starting from a non-directed donor and with length $w_h = |A(h)| \leq L$.

The kidney exchange problem (KEP) is the problem of finding a set of vertex-disjoint cycles of C and chains of H which maximizes the total number of arcs covered by the set.

Several mathematical programming formulations have been proposed for the KEP; see, e.g., Mak-Hau [2017]. For the sake of generality, we are going to assume, here and in the following sections, that KEP is formulated as a 0-1 linear programming problem of the form

$$z(G) = \max \sum_{\ell=1}^{N} a_\ell x_\ell \tag{1}$$

$$\text{subject to } x \in P(G), \tag{2}$$

$$x \in \{0,1\}^N, \tag{3}$$

where x is a vector of binary variables of appropriate length N which expresses what arcs, cycles and chains are used for transplants (the exact interpretation depends on the chosen formulation). The vector of coefficients $a \in \mathbb{R}^N$ reflects the lengths of the cycles and chains, and $P(G) \subseteq \mathbb{R}^N$ is a feasible region defined by a finite list of linear inequalities. (In order to simplify the notations, we do not explicitly state the dependence of $z(G)$ and $P(G)$ on K and L.) In our computational experiments, we have relied on a well-known explicit formulation of KEP, the *Position-Indexed Edge* (PIE) formulation [Dickerson et al., 2016]. This formulation is described in Appendix A.

2.3 Stochasticity in the kidney exchange problem

As explained in Section 1, a solution of (1)-(3) may turn out to be un-implementable. Indeed, a potential transplant that has passed preliminary compatibility tests, and is part of a selected cycle or chain, may not pass the crossmatch test. In order to model this phenomenon, it is customary to introduce, for each arc (i, j), a probability $p_{i,j}$ that the arc passes the crossmatch test, and that the intended transplant can proceed. The events associated with all arcs are assumed to be mutually independent. (A variation of this model occurs when probabilities are associated with vertices, rather than arcs. There is no fundamental distinction and in our computational experiments, we will use vertex probabilities. More generally, one could also imagine a situation where probabilities are specified for subsets of arcs to pass the corresponding crossmatch tests.)

When stochasticity is introduced, it is necessary to specify how the results of the crossmatch tests are used to identify the transplants to be implemented by the exchange program. All non-adaptive strategies share the following generic framework:

- **(Selection)** A subset of arcs, say $T \subseteq A$, is selected for testing.
- **(Testing)** The arcs in T are crossmatched. Let us call R the set of arcs in T that pass the crossmatch test, $R \subseteq T$.
- **(Recourse)** The kidney exchange problem is solved to optimality on the subgraph $G_R = (V, R)$.

In this framework, only the first and third steps are algorithmic: testing is performed by the medical teams, and can be viewed as revealing the value of the random Bernoulli variables associated with the arcs in T. (In Blum et al. [2015] or Assadi et al. [2016], this testing step is replaced by "query" instructions.) The third step can be viewed as providing a *recourse* against the outcome of the testing step, and the selection step is usually implemented so as to maximize the expected value of the solution provided by the recourse, under various restrictions aimed at simplifying the problem.

So, for example, in Dickerson et al. [2013, 2018], the set T is restricted to consist of a collection of pairwise disjoint cycles and chains. In that case, the model (1)-(3) can be used with a suitable re-definition of the objective function coefficients, so as to express the expected number of transplants in each cycle or chain. In this model, the recourse stage is essentially vacuous, since only those cycles that remain after the crossmatch test, and fragments of chains up to the point of failure, will be implemented. In other words, the model assumes *no recourse*. However, as convincingly demonstrated in Pedroso [2014] and Klimentova et al. [2016], solutions from the no-recourse model are overly conservative and do not adequately represent the number of transplants that might actually be performed when allowing a limited number of additional crossmatch tests, see Example 1.

Example 1. *Consider two subgraphs of a given graph G, say $G' = (V', A')$ and $G'' = (V'', A'')$. We say that G' is* embedded *in G'' if $V' \subseteq V''$. Figure 1 displays an example of a 2-cycle $(1-3-1)$ embedded in a 3-cycle $(1-2-3-1)$. Assume that the 3-cycle is identified by the exchange program in the selection stage, but that arc $(1, 2)$ subsequently fails the crossmatch test, whereas arc $(3, 1)$ passes the test. For an exchange program that allows recourse, it would be possible to further test arc $(1, 3)$, with the hope to be able to implement the two transplants $(1, 3)$ and $(3, 1)$.*

The above observation lead Pedroso [2014] and Klimentova et al. [2016] to propose procedures whereby certain types of subgraphs are selected in the first stage, and all arcs embedded in these subgraphs are subsequently crossmatched so as to provide the input for the recourse stage. (As mentioned earlier, in the UK, the NHS has included limited recourse of this nature; see NHS [2017b] or Biró et al. [2019].)

More specifically, Pedroso [2014] proposes to carry out the selection step by solving the deterministic KEP model (1)-(3) with appropriately defined cycle weights $a_c = w'_c$, and next to apply crossmatch tests to all arcs embedded in the selected cycles; this yields a so-called *internal*

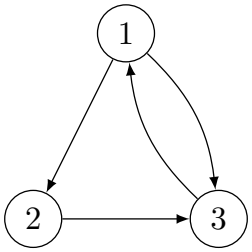


Figure 1: A 3-cycle with an embedded 2-cycle.

recourse. For the model to be correct, the weight w'_c is set equal to the expected optimal value of KEP over the subgraph induced by cycle c. For small enough cycle lengths, these weights can be efficiently computed (see Pedroso [2014]).

Klimentova et al. [2016] extend the previous idea by solving a packing model similar to (1)-(3), but with variables that are associated with subsets of vertices (instead of cycles) of small size; this approach is accordingly called *subset recourse*. More formally, the integer programming model used in subset recourse can be stated as follows. Let Ω denote the set of all *relevant* subsets of vertices (for a precise definition of relevant subsets, we refer to Klimentova et al. [2016]). Define the variables $y_U = 1$ if the vertex subset $U \in \Omega$ is selected for testing, and $y_U = 0$ otherwise. The parameter $a_U = w_U$ denotes now the expected number of transplants that can be realized in the subgraph induced by subset $U \in \Omega$. Then, the selection step of the subset recourse model is formulated as:

$$\max \sum_{U \in \Omega} w_U y_U \tag{4}$$

$$\text{subject to} \sum_{U : v \in U} y_U \leq 1 \qquad \forall\, v \in V, \tag{5}$$

$$y_U \in \{0,1\} \qquad \forall\, U \in \Omega. \tag{6}$$

Klimentova et al. [2016] describe methods that allow them to compute the values of w_U when $|U|$ is not too large.

3 The selection problem

As shown in the previous section, no-recourse, internal recourse and subset recourse models only differ in the limitations that they place on the subsets of vertices that are considered in the selection step, and accordingly, on the coefficients used in the KEP model (1)-(3). The choice of these limitations, however, is rather arbitrary and may needlessly restrict the effectiveness of the procedures.

Our proposal in this paper is to focus instead on the key question, which is in our view: what subset of arcs should be selected and subsequently tested for crossmatch? We intend to answer this question by not restricting ourselves to selecting either disjoint cycles and chains, or small disjoint subsets of vertices in the first step. Instead, we propose a model featuring almost no a priori restrictions on the selection step. Indeed, the only condition we impose is an upper bound on the number of crossmatch tests that can be performed, and hence, on the number of arcs that can be selected. This makes sense, since performing crossmatch tests on all possible transplants within the pool of a kidney exchange program, i.e., testing all arcs of A, is logistically infeasible.

This leads us to the definition of the following problem, called the *selection problem*. Given a directed graph $G = (V, A)$, we denote by $\mathcal{S} = \{A_1, \ldots, A_m\}$ the collection of all subsets of arcs, arbitrarily numbered, with $m = 2^{|A|}$. We interpret each subset $A_s \in \mathcal{S}$ as a possible *scenario*, i.e., as the set of arcs that pass the crossmatch tests under some possible realization of the random

variables. (In the sequel, we will often only use the index s to denote a scenario A_s, and we will write $s \in \mathcal{S}$ instead of $A_s \in \mathcal{S}$.) Thus, for each s, the set $A \setminus A_s$ is the set of arcs that would fail the crossmatch tests. The probability q_s of occurrence of scenario $A_s \in \mathcal{S}$ is computed as

$$q_s := (\prod_{(i,j)\in A_s} p_{i,j})(\prod_{(i,j)\notin A_s} (1 - p_{i,j})).$$

Finally, as mentioned above, we assume that we are given an upper bound B on the number of crossmatch tests we are allowed to perform. The selection problem is now to identify a subset of arcs $T \subseteq A$, with $|T| \leq B$, which maximizes the expected number of transplants in the graph (V, T).

The selection model offers more flexibility than the previous (no-)recourse models, in the sense that every feasible solution of those models yields a feasible solution of the selection problem (provided that the number of arcs to be tested does not exceed B), but not conversely. In fact, for a given budget on the number of arcs to be tested, it is the most flexible model. The difference between an optimal value of the selection problem, and an optimal value of another recourse model indicates how much is lost by restricting the set of feasible solutions, see Example 2 for an illustration.

Example 2. *Figure 2 depicts an instance where the presence of restrictions that are inherent to recourse schemes has a negative impact on the expected number of transplants. Assume that the maximum cycle length is $K = 4$, that no chains are allowed, and that the success probability for each arc (i, j) is $p_{i,j} = p = 0.5$. Assume further that we use the subset recourse procedure where Ω contains all subsets of size at most 4. Then, the optimal solution of* (4)–(6) *consists of the subsets $T_1 = \{a_1, a_2, a_3, a_4\}$ and $T_2 = \{b_1, b_2, b_3, b_4\}$. The testing step requires to crossmatch all 10 arcs induced by T_1 and T_2, which leads to an expected* 1.0625 *transplants. On the other hand, the optimal solution of the selection problem (say, with $B = 10$) would pick instead the 8 arcs $(a_1, a_2), (a_2, a_3), (a_3, a_1), (b_1, b_2), (b_2, b_3), (b_3, b_1), (a_1, b_1), (b_1, a_1)$, and would yield an expected* 1.1328125 *transplants.*

Our formulation of the selection problem is in the same spirit as the models proposed in Blum et al. [2013], Blum et al. [2015], or Assadi et al. [2016]. The main difference is that these authors use local constraints on the number of arcs tested and mostly restrict themselves to 2-cycles.

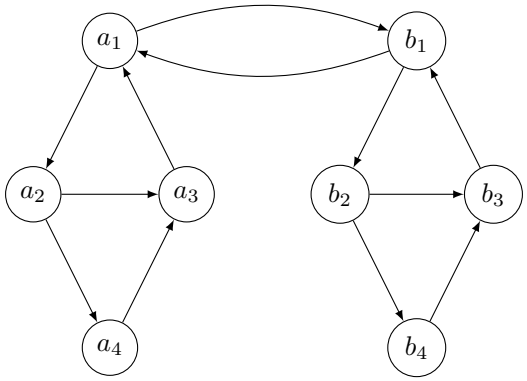


Figure 2: An instance of the selection problem with $K = 4, p = 0.5$

3.1 An integer programming formulation of the selection problem

In order to model the selection problem, we introduce a binary variable $\beta_{i,j}$ for each arc $(i, j) \in A$, with the interpretation that $\beta_{i,j} = 1$ if and only if arc (i, j) is selected to undergo a crossmatch test. Note that for a vector $\beta \in \{0, 1\}^{|A|}$ and for a scenario $A_s \in \mathcal{S}$, the set of arcs R that pass the

crossmatch tests, as introduced in Section 2.3, is exactly $R = R_{s,\beta} = \{(i,j) \in A_s : \beta_{i,j} = 1\}$: in other words, $G_{s,\beta} = (V, R_{s,\beta})$ is the graph on which the recourse step will be performed by solving a deterministic KEP problem.

Let us denote by $x_s \in \{0,1\}^{N_s}$ the binary vector of variables (of appropriate dimension) used to represent a solution to the the kidney exchange problem on the graph $G_s = (V, A_s)$, $s \in S$; we refer to these variables as *scenario* variables. Then, we can extend the KEP formulation (1)-(3) to obtain a generic integer programming formulation of the selection problem:

$$f_{\mathcal{S}} = \max \sum_{s \in \mathcal{S}} q_s \sum_{\ell=1}^{N_s} a_{s,\ell} x_{s,\ell} \tag{7}$$

$$\text{subject to} \sum_{(i,j) \in A} \beta_{i,j} \leq B \tag{8}$$

$$x_s \in P(G_{s,\beta}) \qquad \forall s \in \mathcal{S} \tag{9}$$

$$x_s \in \{0,1\}^{N_s} \qquad \forall s \in \mathcal{S} \tag{10}$$

$$\beta_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{11}$$

The objective function (7) expresses the expected value of a solution to the kidney exchange problem under scenario $s \in S$ by weighing, over all possible scenarios, the quality of this solution by the probability of occurrence of the scenario. Constraint (8) ensures that at most B arcs are selected for crossmatch tests. Next, constraints (9) link the selection of arcs to the choice of transplants to be performed: namely, they ensure that a solution x_s selected for scenario $s \in S$ defines a feasible set of transplants among the arcs that have been submitted to the crossmatch tests (as defined by β) and that have passed them successfully (as expressed by scenario s).

The exact expression of constraints (9) depends on the formulation adopted for KEP. It is usually easy to model (9) by simply adding to the formulation of $P(V, A_s)$ a collection of constraints of the form

$$\sum_{\ell \in Q(s,i,j)} x_{s,\ell} \leq \beta_{i,j} \qquad \forall s \in \mathcal{S}, \forall (i,j) \in A_s, \tag{12}$$

where the set $Q(s,i,j)$ is an appropriate set of indices, forcing some variables $x_{s,\ell}$ to be zero when $\beta_{i,j}$ is zero. In Appendix A, we explicitly state the formulation of the selection problem associated with the PIE formulation of KEP. Note again that similar constraints can be used in conjunction with other formulations of KEP.

3.2 The selection problem for a restricted subset of scenarios

Computationally, the selection problem poses new challenges when compared with the recourse models discussed in Section 2.3. In particular, in approaches that solve those recourse models, all relevant cycles c or subsets U can be explicitly generated if their sizes are sufficiently small. Moreover, the computation of the parameters w'_c or w_U (the expected number of transplants in a subgraph induced by a cycle c or by a subset U) is also facilitated by the fact that the subgraphs under consideration are small (see Pedroso [2014] and Klimentova et al. [2016]). In contrast with these observations, taking into account $|\mathcal{S}| = 2^{|A|}$ different scenarios in model (7)-(11) results in a very large number of decision variables and of constraints.

In order to obtain a tractable model, we therefore restrict our attention to a subset of scenarios, say, $S \subseteq \mathcal{S}$. More precisely, for any subset of scenarios $S \subseteq \mathcal{S}$, we consider a variant of (7)-(11). In this variant, the set $\mathcal{S}$ is replaced by S, and each scenario $s \in S$ is weighted by a factor $\frac{1}{|S|}$ (instead of q_s) in the objective function, i.e., we consider the following variant of the selection

problem:

$$f_S = \max \sum_{s \in S} \frac{1}{|S|} \sum_{\ell=1}^{N_s} a_{s,\ell} x_{s,\ell} \tag{13}$$

$$\text{subject to} \sum_{(i,j) \in A} \beta_{i,j} \leq B \tag{14}$$

$$x_s \in P(G_{s,\beta}) \qquad \forall s \in S \tag{15}$$

$$x_s \in \{0,1\}^{N_s} \qquad \forall s \in S \tag{16}$$

$$\beta_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{17}$$

The model (13)-(17) is referred to as the *restricted selection problem* associated with the subset S of scenarios. If, in addition, constraints (16) are replaced by:

$$0 \leq x_s \leq 1 \qquad \forall s \in S, \tag{18}$$

we call the resulting model the *relaxed restricted selection problem*. By randomly generating the set S in such a way that each $s \in \mathcal{S}$ has a probability q_s of being included in S, we ensure that $\mathbb{E}(f_S) = f_{\mathcal{S}}$. Thus, we expect the optimal value f_S of the restricted selection problem to provide a good approximation of the "true" optimal value $f_{\mathcal{S}}$ of the selection problem when S is large enough. We return to the validity of this assumption in the discussion of our computational experiments (Section 6.2).

In any case, it is crucial to understand that any set of values of the $\beta_{i,j}$ variables satisfying constraint (14) provides a feasible solution of the selection problem and can be implemented in practice. In particular, solving the restricted selection problem over any subset of scenarios always yields a feasible solution to the selection problem.

4 The complexity of the selection problem

Here, we focus on the complexity of the selection problem. We consider three versions: (i) the selection problem with edge probabilities (called SPedge), (ii) the selection problem with vertex probabilities (called SPvertex), and (iii) the selection problem with explicit scenarios (called SPscen). We prove that each of these versions is NP-complete. For reasons of brevity, we give in this section the proof establishing NP-completeness of SPedge; the proofs of NP-completeness for SPvertex and SPscen (which are similar in spirit) can be found in Appendix B. We point out that a result in Blum et al. [2013] establishes the complexity of a related problem: when given edge probabilities and a bound on the number of edges allowed to be incident to each node, they show the resulting problem to be NP-hard. Our hardness results apply to the restricted case of the selection problem where the only allowable cycles are 2-cycles, i.e., $K = 2$, and where we do not allow chains, i.e., $L = 0$. (This is in contrast with the deterministic KEP which can be solved in polynomial time when $K = 2$.) This means that, in this section, we can restrict our attention to compatibility graphs with the property that if arc $(x, y) \in A$, then also $(y, x) \in A$. We model this property by viewing the compatibility graph as an undirected graph consisting of edges instead of arcs. In this view, each edge stands for a 2-cycle of the original directed graph.

Our reductions are from the satisfiability problem SAT, and from a special case of SAT, called (2,2)-3SAT. Recall that an instance of SAT is defined as follows.

Problem: SAT

Instance: A set of n Boolean variables $\{w_1, \ldots, w_n\}$ and a set of m clauses $C = \{c_1, \ldots, c_m\}$ over the variables.

Question: Is there a truth-assignment that satisfies all clauses in C?

The special case (2,2)-3SAT arises when (i) each variable occurs twice negated, and twice unnegated, and (ii) each clause contains exactly three literals. It is proven NP-complete in Berman et al. [2004].

4.1 The selection problem with edge probabilities

Thus, we consider the case where an individual arc in the compatibility graph may, or may not, pass the crossmatch test according to a given probability. In fact, we assume that we are given a probability for each undirected edge; notice that this probability gives the probability that both corresponding directed arcs pass the crossmatch test. Given a random graph $G = (V, E)$, where each edge $e \in E$ has a success probability $p(e)$ to be present (i.e., to pass the crossmatch tests), we denote by $\mathbb{E}(G = (V, E), p)$ the expected value of a maximum matching on $G = (V, E)$.

We now explicitly state the input of the decision version of this selection problem.

Problem: The Decision version of SPedge: DecSPedge

Instance: A simple, undirected graph $G = (V, E)$, numbers $p(e)$ $(0 \leq p(e) \leq 1)$ for each $e \in E$, and numbers B and Z.

Question: Does there exist an edge set $E^* \subseteq E$ such that $|E^*| \leq B$ and $\mathbb{E}((V, E^*), p) \geq Z$?

We will show that SPedge is NP-hard using a reduction from (2,2)-3SAT.

Theorem 1. *DecSPedge is NP-complete, even for $K = 2$ and $L = 0$.*

Proof. Given an instance of (2,2)-3SAT, we construct an instance of the decision version of SPedge as follows. We first build the vertex set V.

- For each clause $c_i \in C$, there are three so-called *clause vertices* $v_{c_{(i,1)}}, v_{c_{(i,2)}}$ and $v_{c_{(i,3)}}$, $i = 1, \ldots, m$.
- For each variable w_j, there are vertices $v_{w_j}, v^+_{w_j}$ and $v^-_{w_j}$, $j = 1, \ldots, n$.

Thus $V = \{v_{c_{(i,1)}}, v_{c_{(i,2)}}, v_{c_{(i,3)}} |\ i = 1, \ldots, m\} \cup \{v_{w_j}, v^+_{w_j}, v^-_{w_j} |\ j = 1, \ldots, n\}$.

We proceed by creating the edge set E. We use three sets of edges: truth-assignment edges (TA), clause-satisfying edges (CS), and dummy edges (D).

- The truth-assignment edges are defined as follows:

  $$TA \equiv \{e(v_{w_j}, v^+_{w_j}), e(v_{w_j}, v^-_{w_j}) |\ j = 1, \ldots, n\}.$$

  Each edge in TA has success probability 1.

- The clause-satisfying edges are defined as follows. Recall that each clause in C contains three variables; we arbitrarily index them using index k, $k = 1, 2, 3$. We have, for each $i = 1, \ldots, m$:

  $$CS^+_i \equiv \cup_j \{e(v_{c_{(i,k)}}, v^+_{w_j}) |\ w_j \text{ occurs unnegated as } k\text{-th variable in clause } c_i, k = 1, 2, 3\},$$

  and

  $$CS^-_i \equiv \cup_j \{e(v_{c_{(i,k)}}, v^-_{w_j}) |\ w_j \text{ occurs negated as } k\text{-th variable in clause } c_i, k = 1, 2, 3\}.$$

  We set $CS^+ = \cup_i CS^+_i$, $CS^- = \cup_i CS^-_i$, and $CS = CS^+ \cup CS^-$. Each edge in CS has success probability $\frac{1}{m}$.

- The dummy edges are defined as follows:

  $$D \equiv \{e(v_{c_{(i,1)}}, v_{c_{(i,2)}}), e(v_{c_{(i,1)}}, v_{c_{(i,3)}}), e(v_{c_{(i,2)}}, v_{c_{(i,3)}}) |\ i = 1, \ldots, m\}.$$

  Each dummy edge has success probability 1. We set $E = TA \cup CS \cup D$.

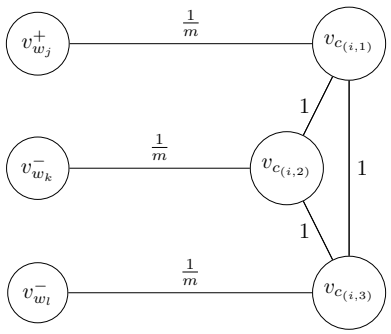


Figure 3: Clause-satisfying edges and dummy edges used in the construction of an instance of SPedge for clause $c_i = (w_j \vee \bar{w_k} \vee \bar{w_l})$.

See Figure 3 for graphical illustration.

Furthermore, we set $B := n+2m$ and $Z := n+m+1-\frac{1}{2m}$, thereby completing the description of the instance of DecSPedge. We claim that there is a satisfying truth-assignment for C if and only if there exists an edge set E^* with $|E^*| \leq B$ and $\mathbb{E}((V, E^*), p) \geq Z$.

$\Rightarrow$ Suppose we have a satisfying truth-assignment for C. We will show how to identify an edge set E^* with $|E^*| \leq n + 2m$ such that the expected value of a maximum matching in $G = (V, E^*)$ is at least $n + m + 1 - \frac{1}{2m}$.

If, in a satisfying truth assignment for C, variable w_j is TRUE, we add edge $e(v_{w_j}, v^-_{w_j})$ to E^*; else, if w_j is FALSE, we add edge $e(v_{w_j}, v^+_{w_j})$ to E^*. Next, for each clause c_i, let w_j be a variable satisfying this clause, and, in fact, let w_j be the k^{th} variable in that clause, $k = 1, 2, 3$. Then, we add edge $e(v_{c_{(i,k)}, v^-_{w_j}})$ to E^* if w_j is TRUE, and we add edge $e(v_{c_{(i,k)}, v^+_{w_j}})$ to E^* if w_j is FALSE. Additionally, we add dummy edges $e(v_{c_{(i,\ell)}}, v_{c_{(i,\ell')}})$ with distinct $\ell, \ell' \neq k$ to E^*, for $i = 1, \ldots, m$.

We consider the three sets of edges and their corresponding contributions to the expected size of a maximum matching.

- Each of the m dummy edges in E^* is not adjacent to any other edge in E^*. Since each dummy edge succeeds with probability 1, they collectively add m to the size of the matching.
- Each of the n truth-assignment edges in E^* is not adjacent to any other edge in E^*. Since each edge in TA succeeds with probability 1, they collectively add n to the size of the matching.
- Next, consider the clause-satisfying edges, of which m are present in E^*. A clause-satisfying edge in E^* can be adjacent to at most one other clause-satisfying edge in E^*; this follows from the fact that the number of clause-satisfying edges that are incident to the same node is bounded by 2, which, in turn, follows from the fact that each literal occurs twice negated and twice unnegated. These edges succeed with probability $\frac{1}{m}$, and since the probability of adjacent edges succeeding is independent, the probability of both edges succeeding is $\frac{1}{m^2}$. Two adjacent edges thus jointly add $\frac{2}{m} - \frac{1}{m^2}$, or $\frac{1}{m} - \frac{1}{2*m^2}$ on average, to the expected size of the matching. All m clause-satisfying edges thus collectively contribute at least $1 - \frac{1}{2m}$ to the size of the matching.

Adding all these contributions together, we conclude that the expected size of the matching constructed is at least $n + m + 1 - \frac{1}{2m}$.

$\Leftarrow$ Suppose there exists an edge set E^* with $|E^*| \leq n + 2m$ such that the expected size of a maximum matching in (V, E^*) equals at least $n + m + 1 - \frac{1}{2m}$. From this we will infer that (i) E^* must contain n edges from TA that prescribe a truth assignment, and contribute n, (ii) E^* must

contain m edges from D contributing m, and (iii) E^* must contain m edges from CS. Moreover, this latter set of edges must contribute at least $1 - \frac{1}{2m}$, implying that none of them is adjacent to either an edge from TA in E^*, or to a dummy edge in E^*. In addition, no pair of CS edges in E^* shares a clause vertex. Thus, there is a CS edge in E^* incident to exactly one of the three vertices $v_{c_{(i,1)}}, v_{c_{(i,2)}}$ and $v_{c_{(i,3)}}$, for each $i = 1, \ldots, m$, implying that the truth assignment found satisfies C.

We now argue the validity of the claims above.

First, we establish that exactly $n + m$ of the edges in E^* are edges from $TA \cup D$, and that no pair of these edges is adjacent. This will imply the existence of a truth assignment, i.e., the edges in TA that are present in E^* prescribe how to set each variable to either TRUE or FALSE. We argue by contradiction.

- Consider the case where less than $n + m$ edges in E^* are from $TA \cup D$. Since every edge in E^* is either an edge from $TA \cup D$ or an edge from CS, it follows that E^* consists of $n + m - q$ edges from $TA \cup D$ and $m + q$ clause-satisfying edges, for some $q \geq 1$. Clearly, the resulting size of a matching in E^* is at most $(n + m - q) + (m + q)\frac{1}{m}$, as a single edge from $TA \cup D$ adds at most 1 to the solution value, while a single edge from CS adds at most $\frac{1}{m}$. For $q \geq 1$, this gives $n + m + 1 - q(\frac{m-1}{m}) < n + m + 1 - \frac{1}{2m}$ (assuming $m \geq 2$).
- On the other hand, if more than $n + m$ edges in E^* are from $TA \cup D$, they can collectively still contribute at most $n + m$ to the size of a matching, as no matching can contain more than n truth-assignment edges and m edges from D. The maximum size of a matching is then bounded by $n + m + (m - q)\frac{1}{m} = n + m + 1 - \frac{q}{m} < n + m + 1 - \frac{1}{2m}$ for $q \geq 1$.
- Finally, two adjacent edges from $TA \cup D$ that are both in E^* collectively only add at most 1 unit to the size of a matching. If $n + m$ edges from $TA \cup D$ are in E^*, and some of these are adjacent, the maximum size of a matching cannot exceed $n + m$.

It follows that E^* contains m edges from D, as well as n edges from TA that prescribe a truth assignment as follows: if $e(w_j, w_j^+)$ ($e(w_j, w_j^-)$ is in E^*, set v_j to FALSE (TRUE), $j = 1, \ldots, n$. Since exactly $n + m$ edges from $TA \cup D$ are in E^*, we know that E^* contains m clause-satisfying edges. It remains to argue that (i) no CS edge in E^* is adjacent to a TA edge in E^*, and (ii) no pair of CS edges in E^* is adjacent.

Since the m clause-satisfying edges in E^* collectively contribute at least $1 - \frac{1}{2m}$ to the expected size of a matching, we derive the following claims.

- We claim that no CS edge in E^* is adjacent to a TA edge in E^*. Indeed, even if one CS edge that is adjacent to a TA edge in E^*, is also in E^*, the contribution of the CS edges in E^* is at most $(m - 1)\frac{1}{m} < 1 - \frac{1}{2m}$. This observation implies that if a CS edge is in E^*, it is consistent with the truth assignment induced by the TA edges, and hence represents a literal satisfying a clause in C.
- If there is a triple of clause nodes $v_{c_{(i,1)}}, v_{c_{(i,2)}}, v_{c_{(i,3)}}$ to which no CS edge in E^* is adjacent, then the contribution from the CS edges to the expected size of the matching does not exceed $(m - 1)\frac{1}{m}$. Since the expected size of a maximum matching exceeds $n + m + 1 - \frac{1}{2m}$, it follows that, for each triple of clause nodes, exactly one must be adjacent to an edge in CS.

These implications ensure that the set of m CS edges present in E^* is consistent with the truth-assignment edges in E^*, thereby satisfying each clause in C.

□

4.2 The selection problem with vertex probabilities

Assume now that each patient-donor pair (i.e., each vertex) in the compatibility graph has a given probability to be able to participate in an exchange. Then, the kidney exchange problem

with *vertex probabilities* becomes relevant. Given a random graph $G = (V, E)$, where each vertex $v \in V$ has a probability $p(v)$ to be present, we denote by $\mathbb{E}(G = (V, E), p)$ the expected value of a maximum matching on $G = (V, E)$. We now state as follows the input of the decision version of the selection problem with vertex probabilities:

Problem: The decision version of SPvertex: DecSPvertex

Instance: A simple, undirected graph $G = (V, E)$, a success probability $p(v)$, $0 \leq p(v) \leq 1$, for each $v \in V$, and numbers B and Z.

Question: Does there exist an edge set $E^* \subseteq E$ such that $|E^*| \leq B$ and $\mathbb{E}((V, E^*), p) \geq Z$?

We state the following result.

Theorem 2. *DecSPvertex is NP-complete, even for $K = 2$ and $L = 0$.*

Proof. See Appendix B. □

4.3 The restricted selection problem with explicit scenarios

We finally define the decision version of the restricted selection problem associated with a subset of scenarios. Given a compatibility graph $G = (V, E)$ with $K = 2$ and $L = 0$, we denote by $z(G)$ the optimal value of this KEP-instance, i.e., the size of a maximum matching on G.

Problem: The decision version of SPscen: DecSPscen

Instance: A simple, undirected graph $G = (V, E)$, a collection of t edge sets $E_s \subseteq E$ $(1 \leq s \leq t)$, numbers B and Z.

Question: Does there exist an edge set $E^* \subseteq E$ such that $|E^*| \leq B$ and $\sum_{s=1}^{t} z(V, E_s \cap E^*) \geq Z$?

In the statement of DecSPscen, the edge sets $E_1, \ldots, E_t$ represent t scenarios. The last inequality in the question implicitly assumes that all scenarios are equally likely, i.e., each scenario has a probability $\frac{1}{t}$ of actually occurring.

We have the following statement.

Theorem 3. *DecSPscen is NP-complete, even for $K = 2$ and $L = 0$.*

Proof. See Appendix B. □

5 Continuous relaxation of the scenarios

Let us now turn to algorithms for the solution of the selection problem (over the complete set $\mathcal{S}$ or over an arbitrary subset S of scenarios: the same discussion applies in both cases, *mutatis mutandis*). In preliminary experiments, we found that most formulations of the (deterministic) KEP, such as PIEF, have very tight linear relaxations. This suggests that replacing the binary scenario variables (say, the generic variables x_s in our formulation (7)-(11) of the selection problem) by their continuous relaxation (18) is unlikely to have a big impact on the arcs that are picked for testing (that is, on the optimal value of the $\beta_{i,j}$ variables); see Section 6.2 for a quantitative assessment of this assumption. Since the vast majority of variables in any formulation of the selection problem are scenario variables, relaxing them decreases the number of binary variables by an order of magnitude. Moreover, when the scenario variables are relaxed, Benders decomposition becomes a natural solution approach for the resulting mixed-integer program: indeed, after fixing the $\beta_{i,j}$ variables to either 0 or 1 in (7)-(11), the problem breaks down into smaller independent subproblems, where each subproblem is the linear relaxation of KEP over the subgraph $G_{s,\beta}$ defined by β and by a specific scenario s.

More precisely, relaxing the scenario variables in the selection problem and using Benders decomposition, we obtain the *master problem*

$$\max \sum_{s\in\mathcal{S}} q_s z_s(\beta) \tag{19}$$

$$\text{subject to} \sum_{(i,j)\in A} \beta_{i,j} \leq B \tag{20}$$

$$\beta_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A, \tag{21}$$

where $z_s(\beta)$ is the optimal value of the following *subproblem*, for each $s \in \mathcal{S}$:

$$z_s(\beta) = \max \sum_{\ell=1}^{N_s} a_{s,\ell} x_{s,\ell} \tag{22}$$

$$\text{subject to } x_s \in P(G_{s,\beta}) \tag{23}$$

$$0 \leq x_{s,\ell} \leq 1 \qquad \forall \ell = 1,\ldots,N_s. \tag{24}$$

Without going into details (we refer to Rahmaniani et al. [2017] for a discussion of Benders decomposition), let us simply mention here that, with constraints (12), solving the dual of (22)-(24) yields valid inequalities of the generic form

$$\sum_{(i,j)\in A_s} u^t_{i,j}\beta_{i,j} + v^t \geq z_s(\beta) \qquad \forall t = 1,\ldots,m, \tag{25}$$

where $u^t_{i,j}$ and v^t are coefficients derived from the extreme points of the dual region. These inequalities can be added to (19)–(21) to obtain the following reformulation of the master problem:

$$\max \sum_{s\in\mathcal{S}} q_s z_s \tag{26}$$

$$\text{subject to} \sum_{(i,j)\in A} \beta_{i,j} \leq B \tag{27}$$

$$\sum_{(i,j)\in A_s} u^t_{i,j}\beta_{i,j} + v^t \geq z_s \qquad \forall t = 1,\ldots,m, \forall s \in \mathcal{S}. \tag{28}$$

$$\beta_{i,j} \in \{0,1\} \qquad \forall (i,j) \in A. \tag{29}$$

Since there is an exponential number of constraints (28), adding them all at once in the model is not efficient. Benders decomposition approaches therefore typically work iteratively, starting at each iteration with a *restricted master problem* of the form (26)–(29), but which only includes a subset of the constraints (28). The value of the β variables in an (optimal) solution of this restricted master problem is then used to formulate the dual of the subproblem (22)-(24) for each scenario $s \in \mathcal{S}$, and to generate new valid inequalities that can be added in the restricted master problem. In our computational experiments, this iterative process is automatically managed by a commercial MIP solver (CPLEX). Before handing the model to the solver, however, we found it useful to strengthen the formulation as explained in the next section.

5.1 Strengthening the formulation

In an optimal solution of the selection problem, each selected arc must be part either of a cycle of length at most K, or of a chain of length at most L: indeed, only such arcs can potentially be used for transplants. In the Benders decomposition of the problem, however, this information is lost to the master problem. In order to palliate this weakness, we add to (7)-(11) (or to (13)-(17)) a list of constraints which force the selected arcs to be part of a cycle or chain. In this way, the solutions of the restricted master problem will display more features of optimal solutions of the full selection problem, even in early iterations of the generation of Benders cuts.

To achieve this goal, we rely on a formulation related to the Position-Indexed Edge formulation of Dickerson et al. [2016] (see Appendix A). Namely, for each vertex ℓ in the graph, a graph copy G^ℓ is created, where G^ℓ is the graph G from which all vertices i with $i < \ell$ have been deleted. Then, new binary variables are created and constrained in such a way, that in graph copy G^ℓ, only arcs leaving vertex ℓ can be in the first position in a cycle or chain. Namely, variable $\phi^\ell_{i,j,k}$ is equal to 1 only if the arc (i,j) is in the k^{th} position of some cycle in graph copy G^ℓ, for all $i,j,\ell \in V$, $i,j \geq \ell$, $(i,j) \in A$, and $k = 1,\ldots,K$. As in Dickerson et al. [2016], the variables $\phi^\ell_{i,j,k}$ can be defined only for $k \in \mathcal{K}(i,j,\ell)$, where the set $\mathcal{K}(i,j,\ell)$ contains those positions in which arc (i,j) can potentially occur in a cycle of G^ℓ. In particular, $\mathcal{K}(i,j,\ell) \subseteq \{1,\ldots,K\}$, $1 \in \mathcal{K}(i,j,\ell)$ only if $i = \ell$, and $K \in \mathcal{K}(i,j,\ell)$ only if $j = \ell$. Similarly, we define variables $\omega_{i,j,k}$ to be equal to 1 only if the arc (i,j) is in the k^{th} position of some chain, for all $(i,j) \in A$, and $k = 1,\ldots,L$. Here again, k can be restricted to a subset $\mathcal{K}(i,j)$ of feasible positions, where $\mathcal{K}(i,j) \subseteq \{1,\ldots,L\}$ and $1 \in \mathcal{K}(i,j)$ only if i is a non-directed donor.

The following constraints can now be added to ensure that each selected arc is part of a cycle or chain of selected arcs:

$$\beta_{i,j} - \sum_{\ell \in V} \sum_{k \in \mathcal{K}(i,j,\ell)} \phi^\ell_{i,j,k} - \sum_{k \in \mathcal{K}(i,j)} \omega_{i,j,k} \leq 0 \qquad \forall (i,j) \in A \quad (30)$$

$$\phi^\ell_{i,j,k} - \beta_{i,j} \leq 0 \qquad \forall \ell \in V, (i,j) \in A, k \in \mathcal{K}(i,j,\ell) \quad (31)$$

$$\omega_{i,j,k} - \beta_{i,j} \leq 0 \qquad \forall (i,j) \in A, k \in \mathcal{K}(i,j) \quad (32)$$

$$\phi^\ell_{i,j,k} - \sum_{(h,i) \in A : (k-1) \in \mathcal{K}(h,i,\ell)} \phi^\ell_{h,i,k-1} \leq 0 \qquad \forall \ell \in V, (i,j) \in A, k \in \mathcal{K}(i,j,\ell), k > 1 \quad (33)$$

$$\phi^\ell_{i,j,k} - \sum_{(j,h) \in A : (k+1) \in \mathcal{K}(j,h,\ell)} \phi^\ell_{j,h,k+1} \leq 0 \quad \forall \ell \in V, (i,j) \in A : j \neq \ell, k \in \mathcal{K}(i,j,\ell), k < K \quad (34)$$

$$\omega_{i,j,k} - \sum_{(h,i) \in A : (k-1) \in \mathcal{K}(h,i)} \omega_{h,i,k-1} \leq 0 \qquad \forall (i,j) \in A, k \in \mathcal{K}(i,j), k > 1. \quad (35)$$

These constraints can be interpreted as follows. Constraints (30)-(32) enforce that an arc can only be selected if it is part of a chain or cycle, whose arcs are also selected. Constraints (33) enforce that, if arc (i,j) is in position k in some cycle of G^ℓ, then there must be a preceding arc in position $k-1$, unless $k = 1$; when $k = 1$, then there is no preceding arc, but the variables $\phi^\ell_{i,j,1}$ are constructed in such a way that i must necessarily be equal to ℓ. Similarly, constraints (34) enforce that (i,j) must have a succeeding arc, unless $j = \ell$ (which ensures that the cycle is completed). Finally, constraints (35) enforce that if an arc (i,j) is chosen in the k^{th} position ($k > 1$) of a chain, then another arc must be chosen in the preceding position. If $k = 1$, there is no preceding position, but the variables are constructed in such a way that the starting vertex of the arc corresponds to a non-directed donor. In this way, the variables $\omega_{i,j,k}$ encode chains in the graph G.

Note that in (30)–(35), contrary to the PIE formulation of Dickerson et al. [2016], we allow selected cycles and chains to overlap, so that several in- or out-going selected arcs may be incident to a same vertex. This is justified by the fact in the recourse step defined in Section 2.3, different cycles or chains may be chosen depending on the observed scenario.

6 Computational experiments

In this section, we discuss the results of various IP-based solution approaches that we implemented to solve the (restricted) selection problem. We first describe the experimental setting in Section 6.1. Then, in Section 6.2, we discuss to what extent (i) restricting the number of scenarios, and (ii) relaxing the integrality of the scenario variables, impacts the quality of a solution. In Section 6.3 we compare the outcomes of our approaches with the results found by recourse approaches from the literature. Finally, in Section 6.4, we discuss the computational efficiency of our approaches when we vary the cycle length and/or admit chains, and when we consider large datasets.

6.1 Experimental setting

For our experiments, we generate kidney exchange graphs using the random generator described in Saidman et al. [2006]. We consider three different graph sizes, consisting respectively of $N = 25$ patient-donor pairs, $N = 50$ patient-donor pairs, and $N = 25$ patient-donor pairs plus one non-directed donor. For each of these sizes, we generate 40 graphs, divided into four groups of 10 graphs, with vertex succes rates p equal to 20%, 40%, 60% and 80%, respectively. These graphs form the basis for all instances. The cycle length K is limited to either 2, or 3, or 4, depending on the experiments, and the maximum chain length L is either 0 or 3 (since these are the most common lengths in real-life kidney exchanges). To complete the instances, we still have to specify the upper bound B on the number of selected arcs. In order to ensure a fair comparison with recourse approaches from literature, we set B equal to the number of arcs selected in a solution found by subset recourse (see Section 2.3) on the same graph for a similar configuration. Specifically, the upper bound for an instance of the selection problem, where the maximum cycle length is K, is equal to the number of arcs tested in the optimal subset recourse solution with cycle length K and with maximum subset size 4. Finally, all instances are tested by including either $|S| = 50$, or 100, or 250, or 500 random scenarios in the formulation (13)-(17) of the restricted selection problem. To be able to compare the solution quality and the computation times of the different algorithmic approaches, each approach uses the exact same scenario set S.

We solve the restricted selection problem using the Position-Indexed Edge (PIE) formulation (see Appendix A) by the following four solution approaches:

1. **BC-IP :** Solve the restricted selection problem (13)-(17) by branch-and-cut.
2. **BC-MIP :** Solve the relaxed restricted selection problem (13)-(15), (17)-(18) by branch-and-cut.
3. **BD-MIP :** Solve the relaxed restricted selection problem (13)-(15), (17)-(18) by Benders decomposition.
4. **BD-MIP+ :** Solve the relaxed restricted selection problem (13)-(15), (17)-(18), together with constraints (30)-(35), by Benders decomposition.

Notice that BC-IP attempts to solve the IP formulation of the problem, whereas BC-MIP, BD-MIP, and BD-MIP+ are solution approaches for different MIP relaxations of the problem. Yet a common feature of solutions of each of these approaches is that a set of arcs is selected, namely the set of arcs (i, j) for which $\beta_{i,j} = 1$ in the respective formulation. In the sequel, when we use the word "solution", we actually refer to this set of arcs.

For each solution approach, the corresponding model is solved using CPLEX 12.8. For the two Benders-based approaches, we rely on CPLEX's built-in Benders functionality. All instances were run with a time limit of two hours on four SandyBridge 2.6Ghz CPUs with 8Gb of memory.

6.2 Justification for solving the (relaxed) restricted selection problem

In this section we investigate the impact on the quality of a solution when (i) we restrict the number of scenarios, and (ii) we relax the integrality of the scenario variables

In Section 3.2, we have introduced the restricted selection problem, where we limit ourselves to a subset S of all possible scenarios; the resulting problem is solved by approach BC-IP. Next, we have suggested the relaxed restricted selection problem, which arises when we relax the integrality of the scenario variables, which we have employed in the three solution approaches BC-MIP, BD-MIP, and BD-MIP+. Both relaxations of the original selection problem simplify the computational problem, but may result in solutions of lower quality compared to the optimum $f_{\mathcal{S}}$. Recall that the quality of a solution is measured by the expected number of transplants that can be realized in a graph that has as an edge-set those edges (i) that are selected, and (ii) which pass the crossmatch test as specified by given probabilities. Optimizing over a restricted subset of scenarios may lead to overfitting, i.e., to the selection of (combinations of) arcs which are disproportionately successful

in the chosen scenarios as compared to the full range of scenarios. On the other hand, relaxing the scenario variables may entail the selection of arcs for which good fractional KEP solutions exist for all scenarios, but no good integer solution.

To asses these potential degradations, we have performed extensive ex-post evaluations of the number of expected transplants associated with the optimal solution β of each formulation. In case an instance was not completely solved within two hours, the best incumbent solution was evaluated (see Section 6.4 for more details about the instances solved). The evaluations were performed in one of two ways. For some of the instances, the expected number of transplants was computed by a modified version of the algorithm employed by Klimentova et al. [2016] for the evaluation of subsets. This procedure allows us to evaluate exactly (ex post) the quality of a solution β. However, due to the presence of large connected components in the graphs which need to be evaluated, this procedure sometimes turned out to be computationally costly. So, for most instances, the solution was evaluated by a Monte Carlo procedure. Namely, given the selection solution β for these instances, we generated a large number of random scenarios, and we used these scenarios to compute an estimate of the expected number of transplants.

		$K=2, L=0$		$K=3, L=0$		$K=4, L=0$		$K=3, L=3$	
p	$\|S\|$	IP	MIP	IP	MIP	IP	MIP	IP	MIP
0.8	50	5.37	5.37	6.85	6.80	7.08	7.08	8.21	8.27
	100	5.38	5.38	6.88	6.88	7.16	7.06	8.29	8.28
	250	5.38	5.36	6.90	6.87	7.17	7.08	8.26	8.31
	500	5.38	5.38	6.91	6.91	7.14	7.08	8.31	8.30
0.6	50	4.06	4.06	4.54	4.55	4.52	4.49	5.74	5.73
	100	4.07	4.06	4.57	4.58	4.58	4.58	5.78	5.78
	250	4.07	4.07	4.62	4.63	4.65	4.65	5.79	5.82
	500	4.08	4.08	4.64	4.64	4.67	4.68	5.85	5.82
0.4	50	2.19	2.19	2.27	2.28	2.23	2.25	2.30	2.28
	100	2.21	2.21	2.32	2.33	2.31	2.31	2.39	2.39
	250	2.21	2.21	2.38	2.38	2.37	2.36	2.44	2.43
	500	2.22	2.22	2.39	2.39	2.38	2.38	2.46	2.46
0.2	50	0.61	0.61	0.59	0.56	0.55	0.54	0.56	0.60
	100	0.63	0.63	0.64	0.64	0.63	0.63	0.67	0.67
	250	0.63	0.63	0.68	0.68	0.68	0.69	0.71	0.71
	500	0.64	0.64	0.69	0.69	0.69	0.69	0.72	0.72

Table 1: Estimate of the expected number of transplants (average over 10 instances), $N = 25$

.

Table 1 reports the results obtained on graphs with $N = 25$ patient-donor pairs and various values of $p, |S|, K, L$. The column labelled 'IP' gives an estimate of the quality of the solution found by using solution approach BC-IP; the value reported is an average over 10 instances. Similarly, the column labelled 'MIP' gives an estimate of the quality of the solution found by using solution approach BD-MIP+; again, the value reported is an average over 10 instances.

First, from Table 1 we see that the effect of varying the number of scenarios depends mostly on the success rate, and to a lesser extent on the maximum cycle/chain length. When $K = 2$ or when $p = 0.8$, the number of scenarios used hardly affects the quality of the solution. However, for $K = 3$ and $L = 0$, enlarging the number of scenarios from 50 to 500 increases the number of expected transplants of the (near-)optimal solution by less than 1 percent for the instances with 80% success rate, while it increases the solution quality by 17 percent for the instances with 20% success rate. So, it appears that for smaller success rates and for larger values of K, L, increasing the number of scenarios matters. The influence of the success rate is further illustrated

by the difference between the objective value of the restricted selection problem (i.e., the expected number of transplants estimated over a limited subset S of scenarios), and the ex-post evaluation over a much larger set. For instances with $K = 3$ and $L = 0$, 50 scenarios and 20% success rate, the accurate ex-post evaluation of the objective value of the restricted selection problem (BC-IP) equals 0.59 (see Table 1). However, the objective function value given by the optimal solution of BC-IP (again averaged over 10 instances) equals 0.82 (this value is not shown in Table 1), which is an overestimation of 44%. By comparison, for instances with $K = 3$, $L = 0$, 50 scenarios and 80% success rate, the average objective function value is 7.04, an overestimation of only 3% over the more accurate value 6.85.

Second, the results clearly show that relaxing the scenario variables has little to no influence on the quality of solutions. Indeed, the value of the solution found by solving BD-MIP+ lies quite close to the value of the solution found by BC-IP. This is an important finding, as we will see that relaxing the scenario variables is computationally quite interesting, while this relaxation does not appear to come at the cost of solution quality.

6.3 Comparison with other approaches

In this section, we compare the quality of the solutions of the (relaxed) restricted selection model found by our approaches, with the quality of the solutions delivered by three recourse approaches from literature. These recourse approaches are: no recourse, internal recourse, and subset recourse; see Section 2.3. More precisely, Table 2 displays the expected number of transplants obtained with each of these three recourse approaches (Column 'Method'), the number of crossmatch tests that the specific recourse approach requires (Column '#Tests'), and the expected number of transplants for the selection problem (Column 'Selection') when given (*a posteriori*) the same number of tests as the recourse approach. Indeed, recall that this number of tests is not *a priori* bounded in a recourse approach, while it is when solving the selection problem.

We conclude from the results depicted in Table 2 that both for internal recourse, and for subset recourse, when $K \geq 3$, solutions found by the selection model dominate the solutions found by these recourse approaches. In fact, as the success rate goes down, the improvement in quality of the solution found by the selection model gets larger with respect to both internal and subset recourse. Especially when compared with internal recourse, the selection model brings significant improvements – ranging up to 12% – at low vertex success probabilities. A closer look at the solutions suggests that this is mostly due to the inability of internal recourse to use overlapping 2-cycles which, at low success rates, provide many more expected transplants per tested arc than longer cycles.

The no recourse model performs very similarly to the selection model, even marginally outperforming it for instances with $L = 0$ and low success rate. This can be explained by the small number of arcs that the no recourse approach uses, making it hard for the selection problem to find better solutions. In these cases, randomness in the scenario set can lead the selection model to select some overlapping 2-cycles, which are inferior to non-overlapping 2-cycles chosen by the no recourse model.

Incidentally, the results in Table 2 confirm that the restriction placed on the number of scenarios and the continuous relaxation of the scenario variables do not deteriorate the solution quality to an unacceptable degree: indeed, the solutions of the selection problem produced under these relaxations are still of higher quality than the solutions of the other models.

6.4 Computational efficiency

In this section, we focus on the computational efficiency of different solution approaches and problem formulations, in relation with properties of the instances.

In a first experiment, we compare all four solution approaches, on graphs of size 25, with maximum cycle length 3 and no chains. In a second experiment, we investigate the impact of the maximum cycle length, by comparing the computation times for cycle lengths 2, 3 and 4 on the same graphs. In a third set of experiments, we allow the use of chains in graphs of size 25+1.

		$K=2, L=0$			$K=3, L=0$			$K=4, L=0$			$K=3,\ L=3$		
	p	Method	Selection	#Tests	Method	Selection	#Tests	Method	Selection	#Tests	Method	Selection	#Tests
No recourse	0.8	4.74	4.74	7.2	5.55	5.56	10.0	5.63	5.76	10.7	6.75	6.74	11.9
	0.6	3.10	3.10	8.6	3.16	3.16	8.9	3.16	3.16	8.9	4.00	4.00	11.8
	0.4	1.50	1.50	9.4	1.52	1.52	9.7	1.52	1.52	9.7	1.49	1.53	10.1
	0.2	0.34	0.33	8.6	0.35	0.33	8.9	0.35	0.33	8.9	0.37	0.41	10.8
Internal recourse	0.8	4.74	4.74	7.2	6.12	6.14	12.5	6.69	6.91	16.5	7.52	7.66	16.1
	0.6	3.10	3.10	8.6	3.90	4.02	14.6	4.27	4.47	19.8	4.89	5.13	19.6
	0.4	1.50	1.50	9.4	1.83	1.94	14.7	2.14	2.31	20.8	1.88	2.05	16.0
	0.2	0.34	0.33	8.6	0.45	0.50	14.5	0.56	0.63	19.8	0.53	0.59	17.1
Subset recourse	0.8	5.32	5.38	15.7	6.78	6.91	19.1	6.99	7.08	19.5	8.16	8.3	23
	0.6	4.02	4.08	20.0	4.60	4.64	22.5	4.62	4.68	22.3	5.73	5.82	28.3
	0.4	2.14	2.22	19.7	2.32	2.39	22.9	2.33	2.38	22.6	2.41	2.46	23.5
	0.2	0.63	0.64	20.2	0.65	0.69	22.9	0.65	0.69	23.1	0.70	0.72	25.3

Table 2: Comparison of expected number of transplants for no recourse, internal recourse, and subset recourse, vs. selection model (BD-MIP, 500 scenarios, $N = 25$)

6.4.1 A comparison of the running times of the solution approaches

In this first experiment, we ran all instances with each solution method (BC-IP, BC-MIP, BD-MIP, BD-MIP+) for cycle length $K = 3$.

p	$\|S\|$	BC-IP	BC-MIP	BD-MIP	BD-MIP+
0.8	50	15	22	518	33
	100	23	23	510	18
	250	149	314	1124	91
	500	812*	813*	1758*	232
0.6	50	12	13	14	8
	100	14	16	18	41
	250	37	48	761	45
	500	154	156	1073	58
0.4	50	11	14	14	10
	100	11	12	12	13
	250	19	21	42	37
	500	31	31	172	36
0.2	50	1.6	4.8	3.0	3.6
	100	2.8	8.4	5.8	5.3
	250	10	10	8.6	7.4
	500	13	17	10	9.2

Table 3: Average computation times (in seconds) over 10 instances for each solution method. $N = 25$, cycle length $K \leq 3$. Each asterisk refers to one instance (out of 10) that did not finish within 2 hours. In such cases, unfinished instances are counted as taking 7200 seconds.

It appears from the results in Table 3 that BD-MIP+ (relaxing the scenario variables, applying Benders decomposition and using the constraints (30)-(35)) generally leads to the lowest computation times, especially for harder instances involving a large number of scenarios or a large success probability. For small numbers of scenarios, we also note the good performance of branch-and-cut on the IP formulations (somewhat surprisingly, in most cases, BC-IP is actually more efficient than its mixed-integer relaxed counterpart BC-MIP). However, we point out that that the quality of the solutions decreases significantly with the number of scenarios (see Section 6.2). Standard Benders decomposition (BD-MIP) does not perform well except for the instances with low success rate. In fact, BD-MIP is generally even slower than BC-IP. On the other hand, adding the constraints (30)-(35) in BD-MIP+ has a very strong, positive impact on the computation time.

6.4.2 Impact of cycle length

In our second set of experiments, we solve the (relaxed) restricted selection problem for different maximum cycle lengths. Average computation times of the two solution approaches BC-IP and BD-MIP+ are displayed in Table 4, with the shortest times in boldface for each combination of instance type and solution method.

From the results in Table 4, we see that the maximum cycle length has a strong influence on the total computation time. Especially in the instances with high success rates, computation times increase sharply with the cycle length. The results also mostly confirm the trend observed in Section 6.4.1, with BD-MIP+ performing better than BC-IP for the harder instances. Interestingly, however, BC-IP remains a competitive alternative when the number of scenarios is not too large.

		$K=2$		$K=3$		$K=4$	
p	$\|S\|$	BC-IP	BD-MIP+	BC-IP	BD-MIP+	BC-IP	BD-MIP+
0.8	50	3.3	**0.2**	**15**	33	153	**117**
	100	3.6	**0.9**	23	**18**	401	**181**
	250	3.9	**3.4**	149	**91**	1156*	**730**
	500	13	**9.0**	812*	**232**	2149**	**1025***
0.6	50	3.0	**0.2**	12	**8.3**	**29**	132
	100	5.0	**1.6**	**14**	41	**40**	52
	250	**6.5**	8.1	**37**	45	266	**228**
	500	**7.6**	25	154	**58**	1502	**936**
0.4	50	3.6	**2.9**	11	**10**	**15**	20
	100	6.1	**2.6**	**11**	13	**12**	15
	250	**7.3**	9.4	**19**	37	**46**	121
	500	**8.4**	11	**31**	37	115	**71**
0.2	50	1.1	**0.1**	**1.6**	3.6	**2.2**	3.9
	100	4.1	**0.0**	**2.8**	5.3	**5.6**	7.0
	250	8.3	**2.3**	10	**7.4**	**11**	14
	500	11	**3.8**	13	**9.2**	17	**14**

Table 4: Average computation times (in seconds) over 10 instances, for $N = 25$ and different maximum cycle lengths. Each asterisk refers to one instance that did not finish within 2 hours. In these cases, unfinished instances are counted as taking 7200 seconds.

6.4.3 The impact of chains

In our third set of experiments, we investigate the impact of the presence of non-directed donors (thereby allowing chains) on the computation times. To set the upper bound on the number of selected arcs, we run subset recourse (see Section 2.3) on a modified version of the graph (since subset recourse was not originally developed to handle chains): namely, for each vertex associated with a patient-donor pair, we add an arc with weight 0 from that vertex to the vertex representing the non-directed donor (referred to as the NDD vertex). We furthermore allow cycles of different lengths, if they start from the NDD vertex.

Table 5 shows that the presence of non-directed donors has a significant impact on computation times. Without a non-directed donor, most instances with cycle length 3 and 80% success rate are solved within 2 hours, even with 500 scenarios. In contrast, the addition of a single non-directed donor results in many instances that could not be solved for the harder configurations. For the instances with 80% and 60% succcess rates, 8 out of 20 instances with 500 scenarios failed to finish, and 5 out of 20 for 250 scenarios. The density of the graph, and as a consequence, the upper bound on the number of selected arcs derived from the subset recourse solution, does play an important role. While 4 out of 10 instances with 80% success rate and 500 scenarios failed to finish in 2 hours, 3 of these instances finished in under 20 seconds. These two groups of instances contain 182 and 122 arcs on average, respectively. The difficulty of the instances with 60% success rate is also apparently increasing with the density of the graphs.

Comparing the different solution approaches, we observe that BD-MIP and BD-MIP+ consistently outperform BC-IP for the harder instances with high success rates. For the easier instances, results are more mixed. We note that unlike in the case of cycle-only instances, BD-MIP is competitive with the other approaches.

p	$\|S\|$	BC-IP	BD-MIP	BD-MIP+
0.8	50	106	69	**57**
	100	196	285	**122**
	250	(9) 2389	(9) 1886	(9) **1748**
	500	(4) 4385	(7) **3283**	(6) 3296
0.6	50	1034	1138	**774**
	100	(7) 2621	**2080**	(7) 2976
	250	(4) 4354	(8) **3379**	(6) 3853
	500	(4) 4409	(4) 4756	(6) **4059**
0.4	50	**9.6**	17.8	28.1
	100	**14**	20	21
	250	451	**438**	742
	500	785	**739**	781
0.2	50	1.3	1.0	**0.3**
	100	**1.1**	**1.1**	2.4
	250	12	9.5	**8.4**
	500	16	13	**11**

Table 5: Average computation time for different solution methods over 10 instances. $N = 25 + 1$ non-directed donor. Maximum cycle length $K = 3$, chain length $L = 3$. Numbers in brackets indicate the number of instances completed in under 2 hours. Unfinished instances are counted as taking 7200 seconds.

6.4.4 Larger datasets

We shortly comment on our computational experience with graphs of size 50. Instances of this size prove mostly too large to solve within the time limit for any approach. For 80% success rate, 6 instances out of 10 fail to solve within the time limit, even for 50 scenarios. For 60% and 40% success rates, this number is 6 and 5 respectively. On the other hand, we are able to solve instances with 20% succes rate, with only 1 out of 10 instances that feature 100 or 250 scenarios failing to finish, and only 2 out of 10 instances that feature 500 scenarios, failing to finish.

7 Conclusions

Selecting the right patient-donor pairs for crossmatching in kidney exchange programs is an important and challenging problem. In this paper, we have introduced a new model for the optimization of kidney exchanges under stochastic failures. We have formulated the resulting problem as a two stage stochastic optimization problem. In terms of the expected number of transplants associated with its optimal solutions, this new model compares favorably with the internal or subset recourse models known from literature. Moreover, we have experimented extensively with different implementations of the model and with different algorithms for solving it. From our experiments, it appears that Benders decomposition applied to a strengthened formulation tends to run faster than the other implementations. As expected, it also turns out that increasing the maximum cycle length, as well as allowing chains, has a strong negative impact on the running times.

The proposals and observations formulated in this paper can be interpreted in at least two ways. On one hand, they show that the selection model can yield exchange plans with a higher expected number of transplants than earlier recourse models. From that perspective, it remains a challenge to develop more efficient methods for solving larger instances of the selection model. On the other hand, the selection model can also be used as a benchmark for assessing the loss incurred when using weaker, but computationally more tractable approaches. Here again, it would be interesting to extend the scope of this comparative analysis to larger instances, so as to better understand the relative merits of various approaches.

Acknowledgments

The authors thank several anonymous reviewers for helpful suggestions on a preliminary conference version of the paper. Bart Smeulders is a postdoctoral fellow of the Belgian F.R.S.-FNRS, whose support is gratefully acknowledged. The research of Frits Spieksma is supported by NWO Gravitation Project NETWORKS, Grant Number 024.002.003.

References

David J Abraham, Avrim Blum, and Tuomas Sandholm. Clearing algorithms for barter exchange markets: Enabling nationwide kidney exchanges. In *Proceedings of the Eigth ACM Conference on Economic Commerce*, pages 295–304. ACM, 2007.

Filipe Alvelos, Xenia Klimentova, Abdur Rais, and Ana Viana. A compact formulation for maximizing the expected number of transplants in kidney exchange programs. In *Journal of Physics: Conference Series*, volume 616, page 012011. IOP Publishing, 2015.

Ross Anderson, Itai Ashlagi, David Gamarnik, Michael Rees, Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. Kidney exchange and the alliance for paired donation: Operations research changes the way kidneys are transplanted. *Interfaces*, 45(1):26–42, 2015.

Sepehr Assadi, Sanjeev Khanna, and Yang Li. The stochastic matching problem with (very) few queries. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 43–60. ACM, 2016.

Soheil Behnezhad, Alireza Farhadi, Mohammad Taghi Hajiaghayi, and Nima Reyhani. Stochastic matching with few queries: New algorithms and tools. *arXiv preprint arXiv:1811.03224v1*, 2018.

Piotr Berman, Marek Karpinski, and Alexander Scott. Approximation hardness of short symmetric instances of MAX-3SAT. *Report IHES-M-2004-28*, 2004.

Péter Biró, Bernadette Haase-Kromwijk, Tommy Andersson, Eyjólfur Ingi Ásgeirsson, Tatiana Baltesová, Ioannis Boletis, Catarina Bolotinha, Gregor Bond, Georg Böhmig, Lisa Burnapp, Katarína Cechlárová, Paola Di Ciaccio, Jiri Fronek, Karine Hadaya, Aline Hemke, Christian Jacquelinet, Rachel Johnson, Rafal Kieszek, Dirk Kuypers, Ruthanne Leishman, Marie-Alice Macher, David Manlove, Georgia Menoudakou, Mikko Salonen, Bart Smeulders, Vito Sparacino, Frits Spieksma, María de la Oliva Valentín Muñoz, Nic Wilson, and Joris van de Klundert. Building kidney exchange programmes in Europe - an overview of exchange practice and activities. *Transplantation*, 103:1514–1522, 2019. doi: 10.1097/TP.0000000000002432.

Avrim Blum, Anupam Gupta, Ariel Procaccia, and Ankit Sharma. Harnessing the power of two crossmatches. In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce*, pages 123–140. ACM, 2013.

Avrim Blum, John P Dickerson, Nika Haghtalab, Ariel D Procaccia, Tuomas Sandholm, and Ankit Sharma. Ignorance is almost bliss: Near-optimal stochastic matching with few queries. In *Proceedings of the Sixteenth ACM Conference on Economics and Computation*, pages 325–342. ACM, 2015.

John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Failure-aware kidney exchange. In *Proceedings of the Fourteenth ACM conference on Electronic Commerce*, pages 323–340. ACM, 2013.

John P Dickerson, David F Manlove, Benjamin Plaut, Tuomas Sandholm, and James Trimble. Position-indexed formulations for kidney exchange. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, pages 25–42. ACM, 2016.

John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Failure-aware kidney exchange. *Management Science*, 2018.

Sommer E Gentry, Robert A Montgomery, and Dorry L Segev. Kidney paired donation: Fundamentals, limitations, and expansions. *American Journal of Kidney Diseases*, 57(1):144–151, 2011.

Xenia Klimentova, João Pedro Pedroso, and Ana Viana. Maximising expectation of the number of transplants in kidney exchange programmes. *Computers & Operations Research*, 73:1–11, 2016.

Vicky Mak-Hau. On the kidney exchange problem: Cardinality constrained cycle and chain problems on directed graphs: A survey of integer programming approaches. *Journal of Combinatorial Optimization*, 33(1):35–59, 2017.

David F Manlove and Gregg Omalley. Paired and altruistic kidney donation in the UK: Algorithms and experimentation. *Journal of Experimental Algorithmics (JEA)*, 19:2–6, 2015.

Robert A Montgomery, Sommer E Gentry, William H Marks, Daniel S Warren, Janet Hiller, Julie Houp, Andrea A Zachary, J Keith Melancon, Warren R Maley, Hamid Rabb, et al. Domino paired kidney donation: A strategy to make best use of live non-directed donation. *The Lancet*, 368(9533):419–421, 2006.

NHS. Annual report on living donor kidney transplantation. Technical report, NHS, 2017a.

NHS. Living donor kidney matching run process. Technical report, NHS, `https://nhsbtdbe.blob.core.windows.net/umbraco-assets-corp/2287/ldkmr-matching-process.pdf`, 2017b. Retrieved 26 April 2018.

Joao Pedro Pedroso. Maximizing expectation on vertex-disjoint cycle packing. In *International Conference on Computational Science and Its Applications*, pages 32–46. Springer, 2014.

Ragheb Rahmaniani, Teodor Gabriel Crainic, Gendreau, and Walter Rei. The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. Kidney exchange. *The Quarterly Journal of Economics*, 119(2):457–488, 2004.

Alvin E Roth, Tayfun Sönmez, M Utku Ünver, Francis L Delmonico, and Susan L Saidman. Utilizing list exchange and nondirected donation through chainpaired kidney donations. *American Journal of Transplantation*, 6(11):2694–2705, 2006.

Susan L Saidman, Alvin E Roth, Tayfun Sönmez, M Utku Ünver, and Francis L Delmonico. Increasing the opportunity of live kidney donation by matching for two-and three-way exchanges. *Transplantation*, 81(5):773–782, 2006.

Appendix A: Integer programming formulations

In Section 2.2, we have introduced a very generic formulation of the deterministic kidney exchange problem, which has been extended in Section 3.1 to a formulation of the selection problem. We present hereunder the specific formulation of the selection problem which has been used in our computational experiments.

Position Indexed Edge formulation

The *Position Indexed Edge* (PIE) formulation is proposed by Dickerson et al. [2016]. The main feature of this formulation is the introduction of *position indices*, which denote in which position a particular arc is used in a cycle. In order to model the selection problem, we further extend this formulation by defining binary variables $\gamma_{i,j,k,\ell,s}$, where $\gamma_{i,j,k,\ell,s} = 1$ if and only if the arc (i,j) is chosen in the k^{th} position of a cycle in graph copy ℓ in scenario s. Arcs in chains are handled similarly, though the graph copies are not needed here: variable $\delta_{i,j,k,s} = 1$ if and only if arc (i,j) is chosen in the k^{th} position of a chain in scenario s. The sets $\mathcal{K}(i,j,\ell,s)$ and $\mathcal{K}'(i,j,s)$ are computed in a pre-processing step, and contain those positions in which (i,j) can be selected in scenario s. We refer to Dickerson et al. [2016] for more details on the pre-computation of these sets.

In the formulation hereunder, the objective function (36) and the constraints (37), (38), (39) are directly inherited from the generic formulation (7), (8), (9), (12). The cycle flow constraints (40) enforce that if an arc arriving at a vertex i is selected in position k in a cycle, another arc leaving that vertex i must be selected in position $k+1$. By not allowing arcs in positions larger than K, the cycle length constraints are enforced. Similarly, the chain flow constraints (41) enforce that an arc may only be chosen in position k of a chain if an arc enters the same vertex in position $k-1$. Graph copies are again used to break symmetry, as in Section 5.1.

$$\max \sum_{s\in S} \sum_{(i,j)\in A_s} \Big(\sum_{\ell\in V} \sum_{k\in\mathcal{K}(i,j,\ell,s)} \gamma_{i,j,k,\ell,s} + \sum_{k\in\mathcal{K}'(i,j,s)} \delta_{i,j,k,s}\Big) \tag{36}$$

$$\text{subject to} \sum_{(i,j)\in A} \beta_{i,j} \le B, \tag{37}$$

$$\sum_{\ell\in V} \sum_{j:(j,i)\in A^\ell_s} \sum_{k\in\mathcal{K}(j,i,\ell,s)} \gamma_{j,i,k,\ell,s} + \sum_{j:(j,i)\in A_s} \sum_{k\in\mathcal{K}'(j,i,s)} \delta_{j,i,k,s} \le 1 \qquad \forall s\in S, i\in V, \tag{38}$$

$$\sum_{\ell\in V} \sum_{k\in\mathcal{K}(i,j,\ell,s)} \gamma_{i,j,k,\ell,s} + \sum_{k\in\mathcal{K}'(i,j,s)} \delta_{i,j,k,s} \le \beta_{i,j} \qquad \forall s\in S, (i,j)\in A_s, \tag{39}$$

$$\sum_{j:((j,i)\in A^\ell_s \wedge k\in\mathcal{K}(j,i,\ell,s))} \gamma_{j,i,k,\ell,s} = \sum_{j:((i,j)\in A^\ell_s \wedge (k+1)\in\mathcal{K}(i,j,\ell,s))} \gamma_{i,j,k+1,\ell,s} \qquad \begin{array}{l}\forall s\in S, \ell\in V,\\ i\in\{\ell+1,\dots,n\},\\ k\in\{1,\dots,K-1\}.\end{array} \tag{40}$$

$$\sum_{j:((j,i)\in A_s \wedge k\in\mathcal{K}'(j,i,s))} \delta_{j,i,k,s} \le \sum_{j:((i,j)\in A_s \wedge k\in\mathcal{K}'(i,j,s))} \delta_{i,j,k-1,s} \quad \forall s\in S, i\in V, k\in\{2,\dots,L\}, \tag{41}$$

$$\beta_{i,j}, \gamma_{i,j,k,\ell,s}, \delta_{i,j,k',s} \in \{0,1\} \quad \forall s\in S, i,j,\ell\in V, k\in\mathcal{K}(i,j,\ell,s), k'\in\mathcal{K}'(i,j,s). \tag{42}$$

Appendix B: The complexity of SPvertex and SPscen

The selection problem with vertex probabilities

Theorem 2. *DecSPvertex is NP-complete, even for $K = 2$ and $L = 0$.*

Proof. Given an instance of (2,2)-3SAT, we construct an instance of DecSPvertex as follows. Let us first build the vertex set V and the associated probabilities as follows; we use a similar, actually simplified, construction as in the proof of Theorem 1.

- For each clause $c_i \in C$, there is a so-called *clause vertex* $v_{c_i} \in V$ with $p(v_{c_i}) = \frac{1}{m}$, $i = 1, \ldots, m$.
- For each variable w_j, there are three vertices $v_{w_j}, v^+_{w_j}$ and $v^-_{w_j}$ in V, each with $p(v_{w_j}) = p(v_{w^+_j}) = p(v_{w^-_j}) = 1$, $j = 1, \ldots, n$.

Thus $V = \{v_{c_i} |\ i = 1, \ldots, m\} \cup \{v_{w_j}, v^+_{w_j}, v^-_{w_j} |\ j = 1, \ldots, n\}$. As in the proof of Theorem 1, we use truth-assignment edges:

$$TA \equiv \{e(v_{w_j}, v^+_{w_j}), e(v_{w_j}, v^-_{w_j}), j = 1, \ldots, n\}.$$

And, we use clause-satisfying edges as follows. For each $i = 1, \ldots, m$ define:

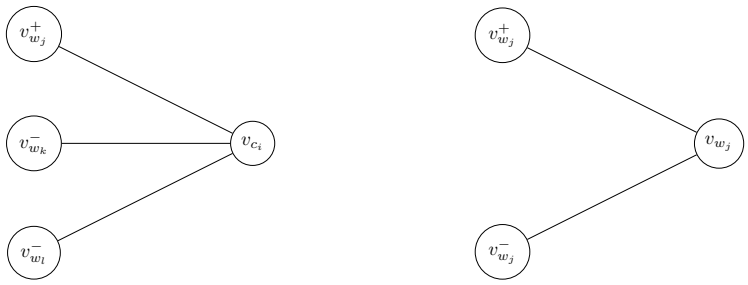
$$CS^+_i \equiv \cup_j \{e(v_{c_i}, v^+_{w_j}) |\ \text{variable } w_j \text{ occurs positively in clause } c_i\},$$

and

$$CS^-_i \equiv \cup_j \{e(v_{c_i}, v^-_{w_j}) |\ \text{variable } w_j \text{ occurs negatively in clause } c_i\}.$$

By setting $E = TA \cup \cup_i CS^+_i \cup \cup_i CS^-_i$, we have constructed G (see Figures 4(a) and 4(b)). Furthermore, we set $B := n + m$, and $Z := n + 1 - \frac{1}{2m}$.

We claim that there is a satisfying truth-assignment for C if and only if there exists an edge set E^* with $|E^*| \leq B$ and $\mathbb{E}((V, E^*), p) \geq Z$. The reasoning we use is similar to the reasoning used in the proof of Theorem 1.



(a) Clause-satisfying edges (b) Truth-assignment edges

Figure 4: Edges used in the construction of an instance of SPvertex.

$\Rightarrow$ Suppose we have a satisfying truth-assignment for C. We will show how to identify an edge set E^* with $|E^*| \leq B = n + m$ such that the expected value of a maximum matching in $G = (V, E^*)$ is at least $n + 1 - \frac{1}{2m}$.

If, in a satisfying truth assignment for C, variable w_j is TRUE, we add edge $e(v_{w_j}, v^-_{w_j})$ to E^*; else we add edge $e(v_{w_j}, v^+_{w_j})$ to E^*, $j = 1, \ldots, n$. In this way, we add n truth-assignment edges to E^*. Further, since C is satisfiable, there exists for each clause c_i, a variable, say w_j, whose truth assignment realizes this clause. For each $i = 1, \ldots, m$, we do the following: if w_j occurs positively in c_i we add edge $e(v_{c_i}, v^+_{w_j})$ to E^*, and if w_j occurs negatively in c_i we add edge $e(v_{c_i}, v^-_{w_j})$ to

E^*. In this way, we add m clause-satisfying edges to E^*. We have now specified E^*; observe that it contains $n+m$ edges. Since each edge in E^* is spanned by two vertices, say v and w with associated success probabilities p_v and p_w, we say that edge $\{v, w\}$ *succeeds* with probability $p_v p_w$. In particular, observe that each edge in TA succeeds with probability 1, whereas each edge in CS succeeds with probability $\frac{1}{m}$.

Consider now the graph (V, E^*). We will identify a matching in this graph, with the required expected size. The matching consists of truth-assignment edges and clause-satisfying edges.

- First, we put all n truth-assignment edges that are in E^* in the matching. Notice that (i) no pair of TA edges in E^* is adjacent (by construction), (ii) no TA edge in E^* is adjacent to a CS edge in E^* (since we have a satisfying truth assignment for C), and (iii) each edge in TA succeeds with probability 1. Thus, the set of TA edges in E^* form a partial matching and jointly contribute expected weight n to the matching.

- Next, consider the m clause-satisfying edges present in E^*. Each clause-satisfying edge is spanned by some node v_{c_i}, and one node from $\{v^+_{w_j}, v^-_{w_j}\}$, say node v. The construction of E^* allows that pairs of CS edges in E^* are adjacent; indeed, it may happen that a same literal makes different clauses true. Observe however, that the reduction from (2,2)-3SAT, where each variable occurs twice negated, and twice unnegated, guarantees that there do not exist three CS edges sharing a node. Thus, there are two cases: either node v is adjacent to a single CS edge in E^*, or node v is adjacent to two CS edges in E^*. In the first case, we simply select the corresponding edge in the matching; it contributes weight $\frac{1}{m}$. In the second case node v is adjacent to two CS edges in E^*; then, we select one of these edges. More precisely, if one such edge succeeds, or if both edges survive, we can select one of them in the matching; the corresponding probability of such an event is $\frac{2}{m} - \frac{1}{m^2}$. Thus, summing over the m edges, we can add at least $1 - \frac{1}{2m}$ to the expected value of the matching.

Concluding, the expected value of this matching equals at least $n + 1 - \frac{1}{2m}$.

$\Leftarrow$ Suppose there exists an edge set E^* with $|E^*| \leq n+m$ such that the expected size of a maximum matching (V, E^*) equals at least $n+1-\frac{1}{2m}$. We construct a truth-assignment satisfying C as follows.

First, we establish that exactly n of the edges in E^* must be truth-assignment edges, and that no pair of these edges is adjacent. This will imply the existence of a truth assignment, i.e., the TA edges present in E^* prescribe how to set each variable to either TRUE or FALSE. By following a similar reasoning as in the proof of Theorem 1, we conclude that E^* contains n edges from TA that prescribe a truth assignment as follows: if $e(w_j, w_j^+)$ ($e(w_j, w_j^-)$ is in E^*, set v_j to FALSE (TRUE), $j = 1, \ldots, n$. Since exactly n truth-assignment edges are in E^*, we know that E^* contains m clause-satisfying edges. It remains to argue that (i) no CS edge in E^* is adjacent to a TA edge in E^*, and (ii) no pair of CS edges in E^* are adjacent to a clause vertex.

Since the m clause-satisfying edges in E^* collectively contribute at least $1 - \frac{1}{2m}$ to the expected size of a matching, we derive the following claims.

- We claim that no CS edge in E^* is adjacent to a TA edge in E^*. Indeed, even if one CS edge that is adjacent to a TA edge in E^*, is also in E^*, the contribution of the CS edges in E^* is at most $(m-1)\frac{1}{m} < 1 - \frac{1}{2m}$. This observation implies that if a CS edge is in E^*, it is consistent with the truth assignment induced by the TA edges, and hence represents a literal satisfying a clause in C.

- If two clause-satisfying edges corresponding to the same clause are both in E^*, they jointly add at most $\frac{1}{m}$ to the expected size of a matching. This is a consequence of the fact that, in that case, these two edges must have some node v_{c_i} in common which - by construction - succeeds with probability $\frac{1}{m}$, while the other nodes spanning these edges do not fail. This ensures that both edges are either both successful or both fail. Thus, if two such edges are in E^*, the contribution of the CS edges in E^* is bounded by $(m-1)\frac{1}{m}$. It follows that no two clause-satisfying edges corresponding to the same clause are in E^*. The m tested

clause-satisfying edges thus all correspond to a different clause, and hence all clauses are satisfied.

These implications ensure that the set of m CS edges present in E^* is consistent with the truth-assignment edges in E^*, thereby satisfying each clause in C. □

The restricted selection problem with explicit scenarios

Theorem 3. *DecSPscen is NP-complete, even for $K = 2$ and $L = 0$.*

Proof. Given an instance of SAT, we construct an instance of DecSPscen as follows. Let us first build the graph $G = (V, E)$. As in the previous proof, we set $V = \{v_{c_i} |\ i = 1, \ldots, m\} \cup \{v_{w_j}, v^+_{w_j}, v^-_{w_j} |\ j = 1, \ldots, n\}$

We now specify the edge sets E_s $(1 \leq s \leq t)$, as well as the set E.

We set $t := m$, i.e., there is a scenario (i.e, an edge set E_s) for every clause c_s. We use, as in previous proofs, the truth-assignment edges that will be present in each set E_s $(1 \leq s \leq t)$:

$$TA := \{e(v_{w_j}, v^+_{w_j}), e(v_{w_j}, v^-_{w_j}), j = 1, \ldots, n\}.$$

In addition, for each clause c_s $(s = 1, \ldots, t)$, define the edge sets

$$CS^+_s := \bigcup_j \{e(v_{c_s}, v^+_{w_j}) |\ \text{variable } w_j \text{ occurs positively in clause } c_s\},$$

$$CS^-_s := \bigcup_j \{e(v_{c_s}, v^-_{w_j}) |\ \text{variable } w_j \text{ occurs negatively in clause } c_s\}.$$

We now define, for each $s = 1, \ldots, t$:

$$E_s := TA \cup CS^+_s \cup CS^-_s.$$

Moreover, we set $E := \cup_s E_s$. Further, we set $B := n + m$ and $Z := m(n + 1)$. This completes the description of an instance of the decision version of SPscen.

We claim that there exists a satisfying truth assignment for C if and only if there exists an edge set $E^* \subseteq E$ with $|E^*| \leq B$ and $\sum_{s=1}^{t} z(V, E_s \cap E^*) \geq m(n + 1)$.

$\Rightarrow$ Suppose we have a satisfying truth-assignment for C. We will show how to identify an edge set E^* with $|E^*| \leq B = n + m$ such that whatever scenario/edge set E_s realizes, at least $n + 1$ edges can be selected in an optimum solution of the resulting instance of the KEP, i.e., $z(V, E_s \cap E^*) \geq n + 1$ for each $s = 1, \ldots, t$.

If, in a satisfying truth assignment for C, variable w_j is TRUE, we add edge $e(v_{w_j}, v^-_{w_j})$ to E^*; else we add edge $e(v_{w_j}, v^+_{w_j})$ to E^*, $j = 1, \ldots, n$. Further, since C is satisfiable, there exists for each clause c_s, a variable, say w_j, whose truth assignment satisfies this clause. Now, for each $s = 1, \ldots, t$, we do the following: if this w_j occurs positively in c_s, we add edge $e(v_{c_s}, v^+_{w_j})$ to E^*, and if this w_j occurs negatively in c_s, we add edge $e(v_{c_s}, v^-_{w_j})$ to E^*. We have now specified E^*; observe that it need not be a matching in G, and that it contains $n + m$ edges.

Consider now the instance of the KEP defined by the graph $(V, E_s \cap E^*)$. We claim that one can always find a matching of size $n + 1$ in $(V, E_s \cap E^*)$. Indeed, there are n edges present in $TA \cap E^*$, and, by construction of E^*, there is one edge incident to v_{c_s} that is in E^*, and is not adjacent to any of the edges in $TA \cap E^*$. Hence $z(V, E_s \cap E^*) \geq n + 1$ for each $s = 1, \ldots, t$, and this implies that the instance of the selection problem with explicit scenarios is a yes-instance.

$\Leftarrow$ Now, we show that if the instance of the decision version of SPscen is a yes-instance, i.e., if there exists an E^* with $|E^*| \leq n + m$ such that $\sum_{s=1}^{t} z(V, E_s \cap E^*) \geq m(n + 1)$, there must be a satisfying truth assignment for C. First, consider the graph $G_s = (V, E_s)$ $(1 \leq s \leq t)$. We claim:

$$z(V, E_s \cap E^*) \leq z(G_s) \leq n+1 \text{ for } s = 1, \ldots, t. \tag{43}$$

Indeed, in an optimum solution to the KEP instance defined by G_s, one can select at most one edge out of the two adjacent edges $\{e(w_j, w_j^+), e(w_j, w_j^-)\}$ for $j = 1, \ldots, n$, and one can select at most one edge incident to node v_{c_s}. Since no other edges exist in E_s, the upper bound in (43) is valid. Further, given that our instance of the decision version of SPscen is a yes-instance, we have:

$$\sum_{s=1}^{t} z(V, E_s \cap E^*) \geq m(n+1). \tag{44}$$

Combining (43) and (44) implies that E^* is such that:

$$z(V, E_s \cap E^*) = n+1 \text{ for each } s = 1, \ldots, t.$$

In words, E^* is such that, when intersected with the edges of any scenario E_s, the size of a maximum matching equals $n+1$. Clearly, this value can only be realized by having in E^* at least one edge out of each of the n pairs $\{e(w_j, w_j^+), e(w_j, w_j^-)\}$ $(1 \leq j \leq n)$, and at least one edge incident to each v_{c_s} $(1 \leq s \leq t)$. However, since we know that $|E^*| \leq n+m$, we conclude that E^* contains exactly one edge from each pair $\{e(w_j, w_j^+), e(w_j, w_j^-)\}$ and exactly one edge incident to each v_{c_s}. The n edges in E^* from the pairs $\{e(w_j, w_j^+), e(w_j, w_j^-)\}$ determine the truth assignment of the variables: if $e(w_j, w_j^+)$ is in E^*, then we set w_j to FALSE, else we set w_j to TRUE. The m edges in E^* incident to the nodes v_{c_s} imply that this truth assignment satisfies C: for each individual clause c_s there is an edge $e(v_{c_s}, v_{w_j})$ in E^*, meaning there is a variable w_j whose truth assignment satisfies clause c_s.

□