Position-Indexed Formulations for Kidney Exchange

JOHN P. DICKERSON, Carnegie Mellon University
DAVID F. MANLOVE, University of Glasgow
BENJAMIN PLAUT, Carnegie Mellon University
TUOMAS SANDHOLM, Carnegie Mellon University
JAMES TRIMBLE, University of Glasgow

A kidney exchange is an organized barter market where patients in need of a kidney swap willing but incompatible donors. Determining an optimal set of exchanges is theoretically and empirically hard. Traditionally, exchanges took place in cycles, with each participating patient-donor pair both giving and receiving a kidney. The recent introduction of chains, where a donor without a paired patient triggers a sequence of donations without requiring a kidney in return, increased the efficacy of fielded kidney exchanges—while also dramatically raising the empirical computational hardness of clearing the market in practice. While chains can be quite long, unbounded-length chains are not desirable: planned donations can fail before transplant for a variety of reasons, and the failure of a single donation causes the rest of that chain to fail, so parallel shorter chains are better in practice.

In this paper, we address the tractable clearing of kidney exchanges with short cycles and chains that are long but bounded. This corresponds to the practice at most modern fielded kidney exchanges. We introduce three new integer programming formulations, two of which are compact. Furthermore, one of these models has a linear programming relaxation that is exactly as tight as the previous *tightest* formulation (which was not compact) for instances in which each donor has a paired patient. On real data from the UNOS nationwide exchange in the United States and the NLDKSS nationwide exchange in the United Kingdom, as well as on generated realistic large-scale data, we show that our new models are competitive with all existing solvers—in many cases outperforming all other solvers by orders of magnitude.

Finally, we note that our *position-indexed chain-edge formulation* can be modified in a straightforward way to take post-match edge failure into account, under the restriction that edges have equal probabilities of failure. Post-match edge failure is a primary source of inefficiency in presently-fielded kidney exchanges. We show how to implement such failure-aware matching in our model, and also extend the state-of-the-art general branch-and-price-based non-compact formulation for the failure-aware problem to run its pricing problem in polynomial time.

Additional Key Words and Phrases: Kidney exchange; matching markets; stochastic matching; integer programming; branch and price

1. INTRODUCTION

Transplantation is the most effective treatment for kidney failure, yet transplant waiting lists have grown rapidly in many countries. In the United States alone, the kidney transplant waiting list grew from $58{,}000$ people in 2004 to $99{,}000$ people in 2014 [Hart et al. 2016]. In order to increase the supply of kidneys for transplant, *kidney exchange schemes* now operate in several countries, including the United States, United Kingdom, Netherlands, and South Korea.

A kidney exchange [Rapaport 1986; Roth et al. 2004] is a centrally-administered barter exchange market for kidneys. If a patient with end-stage renal disease has a

Work supported by NSF grants IIS-1320620 and IIS-1546752, ARO grant W911NF-16-1-0061, Facebook and Siebel fellowships, and by XSEDE through the Pittsburgh Supercomputing Center (Dickerson / Plaut / Sandholm), and by EPSRC grants EP/K010042/1, EP/K503903/1 and EP/N508792/1 (Manlove / Trimble).
Authors' addresses: J.P. Dickerson, B. Plaut, and T. Sandholm, Computer Science Department, Carnegie Mellon University; email: {dickerson,sandholm}@cs.cmu.edu, bplaut@stanford.edu; D.F. Manlove and J. Trimble, School of Computing Science, University of Glasgow; email: david.manlove@glasgow.ac.uk, james.trimble@yahoo.co.uk.

friend or family member who is willing to donate a kidney to him, but unable to do so due to blood- or tissue-type incompatibility, the patient may enter the exchange in the hope of exchanging his donor with the donor of another participating patient. Three-way exchanges are also possible, but most schemes do not carry out four-way or longer exchanges due to the requirement that transplants be carried out simultaneously, and the risk that one of the participants may need to withdraw, in which case the cycle does not go to execution and the pairs go back into the kidney exchange pool.

The scheme administrator carries out periodic "match runs" in which exchanges are selected in order to maximize the number of planned transplants, or a similar goal—perhaps prioritizing pediatric patients, or to those who have been waiting the longest.

A more recent development has been the introduction of *non-directed donors (NDDs)* to kidney exchange schemes [Montgomery et al. 2006; Roth et al. 2006]. An NDD enters the scheme with the intention of donating a kidney, but *without* a paired recipient. Such a donor initiates a *chain*, in which the paired donor of each patient who receives a kidney donates a kidney to the next patient. In contrast to cyclic exchanges, chains can be carried out non-simultaneously, since patients later in the chain are not waiting to be "repaid" for a donation that has already been made. In practice, it is desirable to impose a cap on chain length, since there is an increasing chance that the final exchanges planned for a chain will not proceed as the chain length is increased (due to various reasons such as pre-transplant crossmatch incompatibility, death of a recipient before transplant, the recipient receiving a deceased-donor kidney, and so on).

In many fielded kidney exchanges, an optimal solution is found by using an integer programming (IP) solver to find a set of disjoint cyclic exchanges and chains that maximizes some scoring function. This approach has been tractable so far; Manlove and O'Malley [2015] report that each instance up to October 2014 in the United Kingdom's National Living Donor Kidney Sharing Schemes (NLDKSS)—one of the largest kidney exchange schemes—could be solved in under a second, with similar results using state-of-the-art solvers at other large exchanges in the US [Anderson et al. 2015; Plaut et al. 2016a] and the Netherlands [Glorie et al. 2014]. However, there is an urgent need for faster kidney exchange algorithms, for three reasons:

— Schemes have recently increased chain-length caps, and we expect further increases as more schemes evolve towards using *nonsimultaneous extended altruistic donor (NEAD) chains* [Rees et al. 2009], which can extend across dozens of transplants.
— Opportunities exist for cross-border schemes, which will greatly increase the size of the problem to be solved; indeed, collaborations have already occurred between, for example, the USA and Greece, and between Portugal and Spain.
— The run time for kidney exchange algorithms depends on the problem instance, and is difficult to predict. It is desirable to have improved algorithms to insure against the possibility that future instances will be intractable for current solvers.

With that motivation, this paper presents new scalable integer-programming-based approaches to optimally clearing large kidney exchange schemes, including two models which can comfortably handle chain caps greater than 10.

1.1. Prior research

Substantial prior research helped grow kidney exchange from a thought experiment to its present increasingly-ubiquitous state. We briefly overview the literature from economics, computer science, and operations research.

Theoretical basis for kidney exchange. Roth et al. [2004; 2005; 2007] set the groundwork for large-scale organized kidney exchange. These papers explored what efficient matchings in a steady-state kidney exchange would look like; extensions by Ashlagi et al. [2012], Ashlagi and Roth [2014], and Ding et al. [2015] address shortcomings

in those theoretical models that appeared as kidney exchange became reality. Game-theoretic models of kidney exchange, where transplant centers are viewed as agents with a private type consisting of their internal pools, were presented and explored by Ashlagi and Roth [2014], Toulis and Parkes [2015], and Ashlagi et al. [2015]. Various forms of dynamism, like uncertainty over the possible future vertices in the pool [Ünver 2010; Ashlagi et al. 2013; Akbarpour et al. 2014] or uncertainty over the existence of particular edges [Blum et al. 2013; Dickerson et al. 2013; Blum et al. 2015] have been explored from both an economic and algorithmic efficiency point of view.

Practical approaches to the kidney exchange clearing problem. The two fundamental IP models for kidney exchange are the *cycle formulation*, which includes one binary decision variable for each feasible cycle or chain, and the *edge formulation*, which includes one decision variable for each compatible pair of agents [Abraham et al. 2007; Roth et al. 2007]. In the cycle formulation, the number of constraints is sublinear in the input size, but the number of variables is exponential. In the edge formulation, the number of variables is linear but the number of constraints is exponential. Optimally solving these models has been an ongoing challenge for the past decade.

Constantino et al. [2013] introduced the first two compact IP formulations for kidney exchange, where *compact* means that the counts of variables and constraints are polynomial in the size of the input. Their *extended edge formulation* was shown empirically to be effective in finding the optimal solution where the cycle-length cap is greater than 3, particularly on dense graphs. However, each of the compact formulations introduced by Constantino et al. has a weaker linear program (LP) relaxation than the cycle formulation, even in the absence of NDDs.

The EE-MTZ model [Mak-Hau 2015], another compact formulation, uses the variables and constraints of the extended edge formulation to model cycles and a variant of the Miller-Tucker-Zemlin model for the traveling salesman problem to model chains. The same paper introduces the exponentially-sized SPLIT-MTZ model, which adds redundant constraints to the edge formulation in order to tighten the LP relaxation.

A number of kidney exchange algorithms use the cycle formulation with *branch and price* [Barnhart et al. 1998] to avoid the need to hold an exponential number of variables in memory [Abraham et al. 2007; Dickerson et al. 2013; Glorie et al. 2014; Klimentova et al. 2014; Plaut et al. 2016a]. These have been the fastest algorithms to date for the kidney exchange problem.

An alternative approach to avoiding the cost of keeping the entire model in memory has been constraint generation, using variants of the edge formulation [Abraham et al. 2007]. Anderson et al. [2015] describe an approach based on an algorithm for the prize-collecting traveling salesman problem. This algorithm is particularly effective for solving instances where the cycle-length cap is 3 and there is no cap on the length of chains, but it is outperformed by branch-and-price-based approaches if a finite chain-length cap is used [Plaut et al. 2016a].

Alternative objectives for the kidney exchange problem include maximizing the *expected* number of transplants subject to post-match arc and vertex failures [Dickerson et al. 2013; Pedroso 2014; Alvelos et al. 2015]. Some fielded exchanges use lexicographic optimisation of a hierarchy of objectives [Glorie et al. 2014; Manlove and O'Malley 2015]; we note that our models can be augmented to support this class of objective function.

1.2. Our contribution

This paper introduces three integer program formulations for the kidney exchange problem, two of which are compact. Model size (i.e., memory footprint) often constrains today's kidney exchange solvers; critically, our models are typically much smaller than

the state of the art *while managing to maintain tight linear program relaxations (LPRs)*—which in practice is quite important to proving optimality quickly.

In Section 3, we introduce the *position-indexed edge formulation (PIEF)*, a model for the kidney exchange problem with only cycles that is substantially smaller than, yet has an LPR equivalent to, the model with the tightest LPR for the cycles-only version of the problem [Abraham et al. 2007; Roth et al. 2007]. Section 3 presents the *position-indexed chain-edge formulation (PICEF)* which compactly brings chains into the model via a polynomial number of decision variables; the number of cycle decision variables is exponential in just the maximum *cycle* length (which is typically only 3 or 4 in fielded exchanges). To address that latter exponential reliance on the cycle length, we also present a branch-and-price-based implementation of PICEF.[1]

Throughout, we prove new results regarding the tightness of the LPRs of our models relative to the current state of the art. The tightness of these relaxations hints that our formulations will be competitive in practice; toward that end, we provide extensive experimental evidence that they are. In particular, we show that at least one of PICEF and HPIEF is faster than the best solver from all those provably-optimal solvers contributed in earlier papers that we evaluated in 96.41% of instances considered, with the speed-ups being most evident for larger instance sizes and larger chain caps. In Section 6, we use real and generated data from two nationwide kidney exchange programs—one in the UK, and one in the US—to compare our formulations against other competitive solvers [Abraham et al. 2007; Klimentova et al. 2014; Anderson et al. 2015; Plaut et al. 2016a]. Our new formulations are on par or faster than all other solvers, outperforming all other solvers by orders of magnitude on many problem instances.

Finally, we demonstrate that our PICEF formulation can be adapted straightforwardly to the so-called *failure-aware* model where arcs probabilistically fail after algorithmic match but before transplantation, and we show how to adapt a recent (non-compact) branch-and-price-based formulation due to Dickerson et al. [2013] to perform pricing in polynomial time (Section 6).

2. PRELIMINARIES

Given a pool consisting of patient-donor pairs and non-directed donors (NDDs), we model the kidney exchange problem using a directed graph $D = (V, A)$. The set of vertices $V = \{1, \ldots, |V|\}$ is partitioned into $N = \{1, \ldots, |N|\}$ representing the NDDs and $P = \{|N| + 1, \ldots, |N| + |P|\}$ representing the patient-donor pairs.

For each $i \in N$ and each $j \in P$, A contains the arc (i, j) if and only if NDD i is compatible with patient j. For each $i \in P$ and each $j \in P \backslash \{i\}$, A contains the arc (i, j) if and only if the paired donor[2] of patient i is compatible with patient j. Each arc $(i, j) \in A$ has a weight $w_{ij} \in \mathbb{R}^+$ representing the priority that the scheme administrator gives to that transplant. If the objective is to maximize the number of transplants, each arc has unit weight; most fielded exchanges use weights to encode various prioritization schemes and other value judgments about potential transplants.

Since NDDs do not have paired patients, each vertex representing an NDD has no incoming arcs. Moreover, we assume that no patient is compatible with her own paired donor, and therefore that the digraph has no loops. (The IP models introduced in this

[1] In Appendix C of the full version of the paper [Dickerson et al. 2016], we present the *hybrid position-indexed edge formulation (HPIEF)*, which combines PIEF and PICEF to yield a compact formulation, as well as additional theoretical results and proofs.

[2] In this paper, we assume that each patient has a single paired donor. In practice, a patient may have multiple donors; this can be modeled by regarding such a patient i's vertex in D as representing i and all of her paired donors; an arc (i, j) in D then represents compatibility between at least one of i's donors and patient j. Our model can also handle patients with *no* paired donor by drawing a vertex with no out edges.

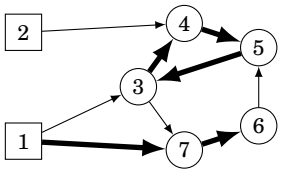

Fig. 1. The directed graph for a kidney-exchange instance with $|N| = 2$ and $|P| = 5$

paper can trivially be adapted to the case where loops exist by adding a binary variable for each loop and modifying the objective and capacity constraints accordingly.)

We use the term *chain* to refer to a path in the digraph initiated by an NDD vertex, and *cycle* to refer to a cycle in the directed graph (which must involve only vertices in P, since vertices in N have no incoming arcs). The weight w_c of each cycle or chain c is defined as the sum of its arc weights.

Given a maximum cycle size K and a maximum chain length L, the *kidney exchange problem* is an optimization problem where the objective is to select a vertex-disjoint packing in D of cycles of size up to K and chains of length up to L that has maximum weight. The problem is NP-hard for realistic parameterizations of K and L [Abraham et al. 2007; Biró et al. 2009]. In practice, K is kept low due to the logistical constraint of scheduling all transplants simultaneously. At both the United Network for Organ Sharing (UNOS) US-wide exchange and the UK's NLDKSS, $K = 3$. The chain cap L is typically longer due to chains being executed non-simultaneously; yet, typically $L \neq \infty$ due to potential matches failing before transplantation. This paper addresses the realistic setting of small cycle cap K and large—but finite—chain cap L.

Figure 1 shows a problem instance with $|N| = 2$ and $|P| = 5$. If each arc has unit weight and $K = L = 3$, then the bold arcs show an optimal solution: cycle $((3,4),(4,5),(5,3))$ and chain $((1,7),(7,6))$, with a total weight of 5.

3. PIEF: POSITION-INDEXED EDGE FORMULATION

3.1. Description of the model

In this section, we present our first of three new IP formulations, the *position-indexed edge formulation* (PIEF). PIEF is a natural extension of the extended edge formulation (EEF) of Constantino et al. [2013]. For this formulation, we assume that the problem instance contains no NDDS.[3]

The PIEF, like the EEF, uses copies of the underlying compatibility digraph D. For each vertex $l \in V$, let $D^l = (V^l, A^l)$ be the subgraph of D induced by $\{i \in V : i \geq l\}$. The PIEF ensures that at most one cycle is selected in each copy, and that a cycle selected in graph copy D^l must contain vertex l.

The first directed graph in Figure 2 is an instance with four patients which we will use as an example in this section. The figure shows graph copies D^1 $(= D)$, D^2, and D^3. The remaining graph copy, $D^4 = (\{4\}, \{\})$, contains no arcs and is not shown.

The main innovation of the PIEF formulation is the use of arc *positions* to index variables; the position of an arc in a cycle is defined as follows. Let $c = (a_1, \ldots, a_{|c|})$ be a cycle represented as a list of arcs in A. Further, assume that we use the unique representation of c such that a_1 leaves the lowest-numbered vertex involved in the cycle. For $1 \leq i \leq |c|$, we say that a_i has position i.

We define $\mathcal{K}(i,j,l)$, the set of positions at which arc (i,j) is permitted to be selected in a cycle in graph copy D^l. For $i,j,l \in V$ such that $(i,j) \in A^l$, let

[3]In Appendix C of the full version of this paper, we present the hybrid PIEF (HPIEF), a compact generalisation of PIEF which can be used for instances with NDDs.

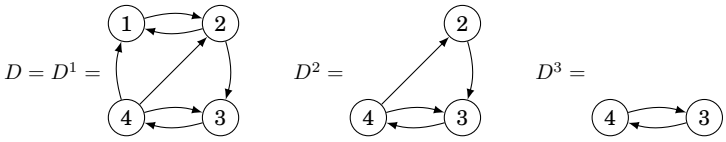

Fig. 2. A kidney exchange instance D with $|N| = 0$ and $|P| = 4$, along with graph copies D^1 ($= D$), D^2, and D^3. $D^4 = (\{4\}, \{\})$ is not shown.

$$\mathcal{K}(i,j,l) = \begin{cases} \{1\} & i = l \\ \{2,\ldots,K-1\} & i,j > l \\ \{2,\ldots,K\} & j = l. \end{cases}$$

Thus, an arc may be selected at position 1 in graph copy l if and only if it leaves vertex l, and any arc selected at position K in graph copy l must enter l.

Now, create a set of binary decision variables as follows. For $i,j,l \in P$ such that $(i,j) \in A^l$, create variable x^l_{ijk} for each $k \in \mathcal{K}(i,j,l)$. Variable x^l_{ijk} takes the value 1 if and only if arc (i,j) is selected at position k of a cycle in graph copy D^l. Returning to our example instance and letting $K = 3$, we give x^1_{342} as an example of a variable in the model; this represents the arc $(3,4)$ being used in position 2 of a cycle in graph copy 1. In full, the set of variables created for this instance is x^1_{121}, x^1_{212}, x^1_{213}, x^1_{232}, x^1_{342}, x^1_{412}, x^1_{413}, x^1_{422}, x^1_{432} (in graph copy D^1), x^2_{231}, x^2_{342}, x^2_{422}, x^2_{423}, x^2_{432} (in graph copy D^2), x^3_{341}, x^3_{432}, and x^3_{433} (in graph copy D^3).

The following integer program finds the optimal cycle packing.

$$\max \quad \sum_{l \in V} \sum_{(i,j) \in A^l} \sum_{k \in \mathcal{K}(i,j,l)} w_{ij} x^l_{ijk} \tag{1a}$$

$$\text{s.t.} \quad \sum_{l \in V} \sum_{j:(j,i) \in A^l} \sum_{k \in \mathcal{K}(j,i,l)} x^l_{jik} \leq 1 \qquad i \in V \tag{1b}$$

$$\sum_{\substack{j:(j,i) \in A^l \wedge \\ k \in \mathcal{K}(j,i,l)}} x^l_{jik} = \sum_{\substack{j:(i,j) \in A^l \wedge \\ k+1 \in \mathcal{K}(i,j,l)}} x^l_{i,j,k+1} \qquad \begin{array}{l} l \in V, \\ i \in \{l+1,\ldots,n\}, \\ k \in \{1,\ldots,K-1\} \end{array} \tag{1c}$$

$$x^l_{ijk} \in \{0,1\} \qquad l \in V, (i,j) \in A^l, k \in \mathcal{K}(i,j,l) \tag{1d}$$

The objective (1a) is to maximize the weighted sum of selected arcs. Constraint (1b) is the capacity constraint for vertices: for each vertex $i \in V$, there must be at most one selected arc whose target is i. Constraint (1c) is the flow conservation constraint. For each graph copy D^l, each vertex i in D^l except the lowest-numbered vertex, and each arc position $k < K$, the number of selected arcs at position k with target i is equal to the number of selected arcs at position $k+1$ with source i. Constraint (1d) ensures that no fractional solutions are selected.[4]

We note that PIEF is not the first IP model for a directed-graph program to use position-indexed variables; Vajda [1961] uses position-indexed variables for subtour elimination in a model for the travelling salesman problem (TSP). Vajda's model is substantially different from the PIEF; most notably, graph copies are not required for Vajda's model because any TSP solution contains exactly one cycle.

[4]Theorem A.3 in Appendix A.1 of the full paper establishes the correctness of the PIEF model.

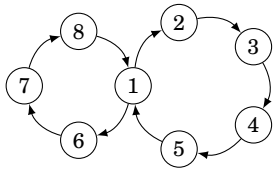

Fig. 3. A graph copy where the reduced PIEF decreases the number of variables in the integer program

3.2. Further reducing the size of the basic PIEF model

We now present methods for reducing the size of the PIEF model while maintaining provable optimality. These reductions are performed as a polynomial-time preprocess (prior to solving the NP-hard kidney exchange clearing problem), and thus may result in practical run time improvements.

3.2.1. Basic reduced PIEF. In typical problem instances, there are many (i, j, k, l) tuples such that x^l_{ijk} takes the value zero in any assignment that satisfies constraints (1b)-(1d). For example, suppose that $K = 4$ and that Figure 3 is graph copy D^1. Arc $(6, 7)$ cannot be chosen at position 3 of a cycle, since the arc only appears in one cycle and it is at position 2 of that cycle. Hence, if we could eliminate variable x^1_{673} from the integer program it would not change the optimal solution. Similarly, all variables for the arc $(3, 4)$ within this graph copy could be eliminated, since this arc does not appear in any cycle of length less than 5.

Following the approach used by Constantino et al. [2013] for the extended edge formulation, we eliminate variables as follows. For $i, j \in V^l$, let d^l_{ij} be the length of the shortest path in terms of arcs from i to j in D^l. For $(i, j) \in A^l$, let

$$\mathcal{K}^{\text{red}}(i, j, l) = \{k : 1 \leq k \leq K \wedge d^l_{li} < k \wedge d^l_{jl} \leq (K - k)\}.$$

For any $k \notin \mathcal{K}^{\text{red}}(i, j, l)$, no cycle in graph copy D^l of length less than or equal to K contains (i, j) at position k, since either there is no sufficiently short path from l to i or there is no sufficiently short path from j to l. Note that $\mathcal{K}^{\text{red}}(i, j, l) \subseteq \mathcal{K}(i, j, l)$. We can substitute $\mathcal{K}^{\text{red}}(i, j, l)$ for $\mathcal{K}(i, j, l)$ in (1a)-(1d), yielding a smaller integer program—*PIEF-reduced (PIEFR)*—with the same optimal solution.

3.2.2. Elimination of variables at position 1 and K: the PIEFR^2 formulation. In the PIEFR model with $K \geq 3$, variables at position 1 are redundant, since $x^l_{lj1} = 1$ if and only if $x^l_{ji2} = 1$ for some i. Similarly, variables at position K are redundant; $x^l_{jlK} = 1$ if and only if $x^l_{ij(K-1)} = 1$ for some i. We can eliminate variables at positions 1 and K from PIEFR as follows. Define a modified weight function w': for all $i, j, l \in P$ such that $(i, j) \in A^l$ and all $k \in \{2, \ldots, K - 1\}$, let

$$w'(i, j, k, l) = \begin{cases} w_{ij} + w_{li} & k = 2 \\ w_{ij} + w_{jl} & k = K - 1 \\ w_{ij} & \text{otherwise.} \end{cases}$$

With this weight function, a selected arc (i, j) at position 2 of a cycle in D^l contributes to the objective value its own weight plus the weight of the implicitly selected arc (l, i). An arc (i, j) at position $K-1$ of a cycle in D^l contributes its own weight plus the weight of of the implicitly selected arc (j, l).

For $i, j, l \in P$ such that $(i, j) \in A^l$, define the restricted set of permitted arc positions:

$$\mathcal{K}^{\text{red}^2}(i,j,l) = \mathcal{K}^{\text{red}}(i,j,l) \setminus \{1, K\}.$$

For $i,j,l \in P$ such that $(i,j) \in A^l$, and for each $k \in \mathcal{K}^{\text{red}^2}(i,j,l)$, create a binary variable x^l_{ijk} . The following IP, denoted PIEFR^2 (PIEF reduced twice), is solved.

$$\max \quad \sum_{l \in V} \sum_{(i,j) \in A^l} \sum_{k \in \mathcal{K}^{\text{red}^2}(i,j,l)} w'(i,j,k,l) x^l_{ijk} \quad \text{subject to} \tag{2a}$$

$$\sum_{l \in V} \left(\sum_{j:(j,i) \in A^l} \sum_{k \in \mathcal{K}^{\text{red}^2}(j,i,l)} x^l_{jik} + \sum_{\substack{j:(i,j) \in A^l \wedge \\ 2 \in \mathcal{K}^{\text{red}^2}(i,j,l)}} x^l_{ij2} \right) + \sum_{\substack{h,j:j \neq i \wedge \\ (h,j) \in A^i \wedge \\ K-1 \in \mathcal{K}^{\text{red}^2}(h,j,i)}} x^i_{hj(K-1)} \leq 1 \quad i \in V \tag{2b}$$

$$\sum_{\substack{j:(j,i) \in A^l \wedge \\ k \in \mathcal{K}^{\text{red}^2}(j,i,l)}} x^l_{jik} = \sum_{\substack{j:(i,j) \in A^l \wedge \\ k+1 \in \mathcal{K}^{\text{red}^2}(i,j,l)}} x^l_{i,j,k+1} \quad \begin{array}{l} l \in V, \\ i \in \{l+1, \ldots, n\}, \\ k \in \{2, \ldots, K-2\} \end{array} \tag{2c}$$

$$x^l_{ijk} \in \{0,1\} \quad \begin{array}{l} l \in V, (i,j) \in A^l, \\ k \in \mathcal{K}^{\text{red}^2}(i,j,l) \end{array} \tag{2d}$$

The constraints of PIEFR^2 differ from those of PIEFR (1b-1d) in the following two respects. First, the second term in parentheses in the PIEFR^2 capacity constraint for vertex i (2b) ensures that any selected arc (i,j) at position 2 of a cycle in D^l counts towards the capacity for i, since it is implicit that the arc (l,i) is also chosen. The final term on the left-hand side of constraint (2b) serves the same function for selected arcs at position $K-1$, since an arc at position K is implicitly chosen. Second, the flow conservation constraint (2c) is not required for $k \in \{1, K-1\}$, since arcs at positions 1 and K are not modelled explicitly in PIEFR^2.

3.2.3. Vertex-ordering heuristic. We can reduce the number of variables in the reduced PIEF model by carefully choosing the order of vertex labels in the digraph D. We have found that relabelling the vertices in descending order of total degree is an effective heuristic to this end. To estimate the effect of this ordering heuristic on model size, we generated the PIEFR^2 model for each of the ten PrefLib instances [Mattei and Walsh 2013] with 256 vertices and no NDDs. The heuristic reduced the variable count by a mean of 38 percent, and reduced the constraint count by a mean of 60 percent.

3.3. PIEF has a tight LPR

We now compare the LPR bound of PIEF to those of other popular IP models. The tightness of an LPR is typically viewed as a proxy for how well an IP model will perform in practice, due to the important role the relaxation plays in modern branch-and-bound-based tree search. In this section, we compare the LPR of PIEF against the IP formulation with the tightest LPR, the *cycle formulation*. While the number of decision variables in the cycle formulation model is exponential in the cycle cap K, PIEF maintains an LPR that is just as tight, but has far fewer variables if $K > 3$.

3.3.1. Cycle formulation. We begin by reviewing the cycle formulation due to Abraham et al. [2007] and Roth et al. [2007], and note that the formulation is equivalent to that due to Anderson et al. [2015] if chains are disallowed. In the cycle formulation, a

binary variable z_c is used for each feasible cycle or chain c to represent whether it is selected in the solution. For each vertex v, there is a constraint to ensure that v is in at most selected cycle or chain. Let $\mathcal{C}$ be the set of cycles in D of length at most K and let $\mathcal{C}'$ be the set of NDD-initiated chains in D of length at most L. The model to be solved is as follows.

$$\max \sum_{c \in \mathcal{C} \cup \mathcal{C}'} w_c z_c \tag{3a}$$

$$\text{s.t.} \sum_{c \in \mathcal{C} \cup \mathcal{C}' : i \text{ appears in } c} z_c \leq 1 \qquad i : 1 \leq i \leq |V| \tag{3b}$$

$$z_c \in \{0, 1\} \qquad c \in \mathcal{C} \cup \mathcal{C}' \tag{3c}$$

Constraints (3b) ensure that the selected cycles and chains are vertex disjoint.

3.3.2. LPR of PIEF. We now show that the LPR of PIEF is exactly as tight as that of the cycle formulation. Formally, if A and B are two IP formulations for the kidney exchange problem, we write that A *weakly dominates* B, denoted $Z_A \preceq Z_B$, if for every problem instance, the LPR objective value under A is no greater than the LPR objective value under B. Further, we say that A *strictly dominates* B, denoted $Z_A \prec Z_B$, if $Z_A \preceq Z_B$ and for some problem instance, the LPR objective value under A is strictly smaller than the LPR objective value under B. Finally, we write $Z_A = Z_B$ if $Z_A \preceq Z_B$ and $Z_B \preceq Z_A$.

THEOREM 3.1. *$Z_{CF} = Z_{PIEF}$ (without chains).*

4. PICEF: POSITION-INDEXED CHAIN-EDGE FORMULATION

4.1. Description of the model

Our second new IP formulation, PICEF, uses a variant of PIEF for chains, and—like the cycle formulation—uses one binary variable for each cycle. The idea of using variables for arcs in chains and a variable for each cycle was introduced in the PC-TSP-based algorithm of Anderson et al. [2015]. The innovation in our IP model is the use of position indices on arc variables, which results in polynomial counts of constraints and arc-variables; this is in contrast to the exponential number of constraints in the PC-TSP-based model.

Unlike PIEF, PICEF does not require copies of D. Intuitively, this is because a chain is a simpler structure than a cycle, with no requirement for a final arc back to the initial vertex.

We define $\mathcal{K}'(i,j)$, the set of possible positions at which arc (i,j) may occur in a chain in the digraph D. For $i, j \in V$ such that $(i,j) \in A$,

$$\mathcal{K}'(i,j) = \begin{cases} \{1\} & i \in N \\ \{2, \ldots, L\} & i \in P \end{cases}.$$

Thus, any arc leaving an NDD can only be in position 1 of a chain, and any arc leaving a patient vertex may be in any position up to the cycle-length cap L, except 1.

For each $(i,j) \in A$ and each $k \in \mathcal{K}'(i,j)$, create variable y_{ijk}, which takes value 1 if and only if arc (i,j) is selected at position k of some chain. For each cycle c in D of length up to K, define a binary variable z_c to indicate whether c is used in a packing.

For example, consider the instance in Figure 4, in which $|N| = 2$ and $|P| = 4$. Suppose that $K = 3$ and $L = 4$, and suppose further that each arc has unit weight. The IP model includes variables y_{131}, y_{141}, and y_{241}, corresponding to arcs leaving NDDs. For each $k \in 2, 3, 4$, the model includes variables y_{34k}, y_{45k}, y_{56k}, y_{64k}, and y_{65k}, corresponding to arcs between donor-patient pairs at position k of a chain. Finally, the model includes z_c variables for the cycles $((4,5),(5,6),(6,4))$ and $((5,6),(6,5))$.

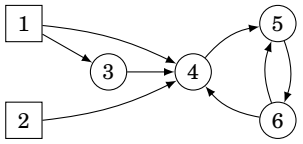

Fig. 4. An instance with $|N| = 2$ and $|P| = 4$

The following IP is solved to find a maximum-weight packing of cycles and chains.

$$\max \quad \sum_{(i,j)\in A} \sum_{k\in\mathcal{K}'(i,j)} w_{ij} y_{ijk} + \sum_{c\in\mathcal{C}} w_c z_c \tag{4a}$$

$$\text{s.t.} \quad \sum_{j:(j,i)\in A} \sum_{k\in\mathcal{K}'(j,i)} y_{jik} + \sum_{c\in\mathcal{C}: i \text{ appears in } c} z_c \leq 1 \qquad i \in P \tag{4b}$$

$$\sum_{j:(i,j)\in A} y_{ij1} \leq 1 \qquad i \in N \tag{4c}$$

$$\sum_{\substack{j:(j,i)\in A \wedge \\ k\in\mathcal{K}'(j,i)}} y_{jik} \geq \sum_{j:(i,j)\in A} y_{i,j,k+1} \qquad \begin{array}{l} i \in P, \\ k \in \{1,\ldots,K-1\} \end{array} \tag{4d}$$

$$y_{ijk} \in \{0,1\} \qquad (i,j) \in A, k \in \mathcal{K}'(i,j) \tag{4e}$$

$$z_c \in \{0,1\} \qquad c \in \mathcal{C} \tag{4f}$$

Inequality (4b) is the capacity constraint for patients: each patient vertex is involved in at most one chosen cycle or incoming arc of a chain. Inequality (4c) is the capacity constraint for NDDs: each NDD vertex is involved in at most one chosen outgoing arc. The flow inequality (4d) ensures that patient-donor pair vertex i has an outgoing arc at position $k+1$ of a selected chain only if i has an incoming arc at position k; we use an inequality rather than an equality since the final vertex of a chain will have an incoming arc but no outgoing arc.[5]

We now give an example of each of the inequalities (4b–4d) for the instance in Figure 4. For $i = 4$, the capacity constraint (4b) ensures that $y_{141} + y_{241} + y_{342} + y_{343} + y_{344} + z_{((4,5),(5,6),(6,4))} \leq 1$. For $i = 1$, the NDD capacity constraint (4c) ensures that $y_{131} + y_{141} \leq 1$. For $i = 5$ and $k = 2$, the chain flow constraint (4d) ensures that $z_{452} + z_{652} \geq z_{563}$; that is, the outgoing arc $(5,6)$ can only be selected at position 3 of a chain if an incoming arc to vertex 5 is selected at position 2 of a chain.

In our example in Figure 4, the optimal objective value is 4. One satisfying assignment that gives this objective value is $y_{131} = y_{342} = z_{((5,6),(6,5))} = 1$, with all other variables equal to zero.

4.2. Practical implementation of the PICEF model

We now discuss methods for the practical implementation of PICEF, first by reducing the number of decision variables via a polynomial-time preprocess, and second by tackling the large number of decision variables for cyclic exchanges via a branch-and-price-based transformation of the model.

4.2.1. Reduced PICEF. We can reduce the PICEF model using a similar approach to the PIEF reduction in Subsection 3.2.1. For $i \in P$, let $d(i)$ be length of the shortest path in terms of arcs from some $j \in N$ to i. Since any outgoing arc from i cannot

[5]Theorem B.4 in Appendix B.1 of the full paper establishes the correctness of the PICEF model.

appear at position less than $d(i)+1$ in a chain, we can replace $\mathcal{K}'$ in PICEF with $\mathcal{K}^{\text{red}}$, defined as follows:

$$\mathcal{K}^{\text{red}}(i,j) = \begin{cases} \{1\} & i \in N \\ \{d(i)+1,\ldots,L\} & i \in P. \end{cases}$$

4.2.2. A branch and price implementation of PICEF. We now discuss a method for scaling PICEF to graphs with high cycle caps, or large graphs with many cycles; this method maintains the full set of arc decision variables, but only incrementally considers those corresponding to cycles.

Formally, for $V = P \cup N$, the number of cycles of length at most K is $O(|P|^K)$, making explicit representation and enumeration of all cycles infeasible for large enough instances. With one decision variable per cycle, Abraham et al. [2007] could not even write the full integer program in memory for instances as small as 1000 pairs.

Branch and price is a method where only a subset of the decision variables are kept in memory, and columns (in the case of PICEF, only those corresponding to cycle variables) are slowly added until correctness is proven at each node in a branch-and-bound search tree. If necessary, superfluous columns can also be removed from the model, in order to prevent its size from exceeding memory.

The following process occurs at each node in the search tree: first, the LP relaxation of the current model (which may contain only a small number of cycles) is solved. The next step is to generate *positive price cycles*: cycles that have the potential to improve the objective value if included in the model.

The price of a cycle c is given by $\sum_{(i,j)\in c}(w_{ij}-\delta_i)$, where δ_i is the dual value of vertex i. While there exist any positive price cycles at a node in the search tree, optimality of the reduced LP has not yet been proved at that node. The *pricing problem* is to bring at least one new positive price cycle into the model, or prove that none exist. Multiple methods exist for solving the pricing problem in kidney exchange [Abraham et al. 2007; Glorie et al. 2014; Mak-Hau 2015; Plaut et al. 2016a]; in our experimental section, we use the cycle pricer of Glorie et al. [2014] with the bugfix of Plaut et al. [2016a].

Once no more positive price cycles exist, the reduced LP at a specific node is guaranteed to be optimal. However, it may not be integral: in this case, branching occurs, as in standard brand-and-bound tree search. In our experiments, we explore branches in depth-first-order unless optimality is proven at all nodes in the search tree.

4.3. The LPR of PICEF is not as tight

As an analogue to Section 3, we now compare the LPR of PICEF against the cycle formulation LPR. Unlike in the PIEF case, where Theorem 3.1 showed an equivalence between the two models' relaxations, we show that PICEF's relaxation can be looser than that of the cycle formulation. Theorem 4.1 gives a simple construction showing this, while Theorem 4.2 presents a family of graphs on which PICEF's LPR is arbitrarily worse than that of the cycle formulation.

THEOREM 4.1. *$Z_{CF} \prec Z_{PICEF}$ (with chains).*

Indeed, Theorem 4.2 shows that the ratio between the optimum objective value for the relaxations of PICEF and the cycle formulation can be made arbitrarily large.

THEOREM 4.2. *Let $z \in \mathbb{R}^+$ be given. There exists a problem instance for which $Z_{PICEF}/Z_{CF} > z$, where Z_{PICEF} is the objective value of the LPR of PICEF and Z_{CF} is the objective value of the LPR of the cycle formulation.*

While the results of Theorems 4.1 and 4.2 may be disheartening, in the following section, we give experimental evidence that PICEF (as well as its branch-and-price-based interpretation) perform extremely competitively on real and generated data.

5. EXPERIMENTAL COMPARISON OF STATE-OF-THE-ART KIDNEY EXCHANGE SOLVERS

In this section, we compare implementations of our new models against existing state of the art kidney exchange solvers. To ensure a fair comparison, we received code from the author of each solver that is not introduced in this paper. We compare run times of the following state-of-the-art solvers:

— BNP-DFS, the original branch-and-price-based cycle formulation solver due to Abraham et al. [2007];
— BNP-POLY, a branch-and-price-based cycle formulation solver with pricing due to Glorie et al. [2014] and Plaut et al. [2016a];[6]
— CG-TSP, a recent IP formulation based on a model for the prize-collecting traveling salesman problem, with constraint generation [Anderson et al. 2015];
— PICEF, the model from Section 4 of this paper;
— BNP-PICEF, a branch and price version of the PICEF model, as presented in Section 4.2.2 of this paper;
— HPIEF, the Hybrid PIEF model from Appendix C of the full version of this paper (which reduces to PIEF for $L = 0$); and
— BNP-DCD, a branch-and-price algorithm using the Disaggregated Cycle Decomposition model, which is related to both the cycle formulation and the extended edge formulation [Klimentova et al. 2014].

A cycle-length cap of 3 and a time limit of 3600 seconds was imposed on each run. When a timeout occurred, we counted the run-time as 3600 seconds.

We test on two types of data: real and generated. Section 5.1 shows run time results on *real* match runs, including 286 runs from the UNOS US-wide exchange, which now contains 143 transplant centers, and 17 runs from the NLDKSS UK-wide exchange, which uses 24 transplant centers. Section 5.2 increases the size and varies other traits of the compatibility graphs via a realistic generator seeded by the real UNOS data. We find that PICEF and HPIEF substantially outperform all other models.

5.1. Real match runs from the UK- and US-wide exchanges

We now present results on real match run data from two fielded nationwide kidney exchanges: The United Network for Organ Sharing (UNOS) US-wide kidney exchange where the decisions are made by algorithms and software from Prof. Sandholm's group, and the UK kidney exchange (NLDKSS) where the decisions are made by algorithms and software from Dr. Manlove's group.[7] The UNOS instances used include all the match runs starting from the beginning of the exchange in October 2010 to January 2016. The exchange has grown significantly during that time and chains have been incorporated. The match cadence has increased from once a month to twice a week; that keeps the number of altruists relatively small. On average, these instances have $|N| = 2$, $|P| = 231$, and $|A| = 5021$. The NLDKSS instances cover the 17 quarterly match runs during the period January 2012–January 2016. On average, these instances have $|N| = 7$, $|P| = 201$, and $|A| = 3272$.

Figure 5 shows mean run times across all match runs for both exchanges; Appendix E in the full version of the paper gives additional statistics like minimum and maximum run times, as well as their standard deviations. Immediately obvious is that

[6]Recently, Plaut et al. [2016b] showed a correctness bug in both implementations of the BNP-POLY-style solvers due to Glorie et al. [2014] and Plaut et al. [2016a]; for posterity, we still include these run times. Furthermore, we note that the objective values returned by BNP-POLY always equaled that of the other provably-correct solvers on all of our test instances.

[7]Due to privacy constraints on sharing real healthcare data, the UNOS and NLDKSS experimental runs were necessarily performed on different computers—one in the US and one in the UK. All runs *within* a figure were performed on the same machine, so relative comparisons of solvers within a figure are accurate.

the non-compact formulations—BNP-DFS and CG-TSP—tend to scale poorly compared to our newer formulations. Interestingly, BNP-PICEF tends to perform worse than the base PICEF and HPIEF; we hypothesize that this is because branch-and-price-based methods are necessarily more "heavyweight" than standard IP techniques, and the small size of presently-fielded kidney exchange pools may not yet warrant this more advanced technique. Perhaps most critically, both PICEF and HPIEF clear real match runs in both exchanges within seconds.

In the NLDKSS results, the wide fluctuation in mean run time as the chain cap is varied can be explained by the small sample size of available NLDKSS instances, and the fact that the algorithms other than HPIEF and PICEF occasionally timed out at one hour. By contrast, each of the HPIEF and PICEF runs on NLDKSS instances took less than five seconds to complete. We also note that the LP relaxation of PICEF and HPIEF are very tight in practice; the LPR bound equaled the IP optimum for 614 of the 663 runs carried out on NLDKSS data.

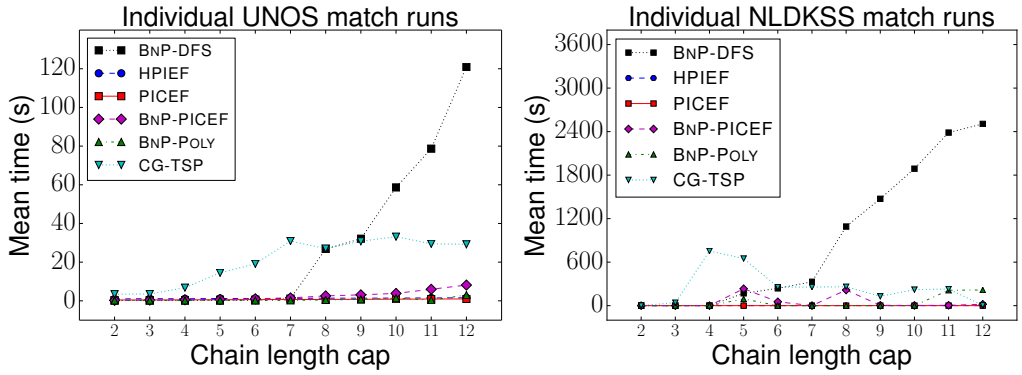

Fig. 5. Mean run times for various solvers on 286 real match runs from the UNOS exchange (left), and 17 real match runs from the UK NLDKSS exchange (right).

We remark that the BNP-DCD model due to Klimentova et al. [2014] was run on all NLDKSS instances where the chain cap L was equal to 0. Larger values of L could not be tested since the current implementation of the model in our possession does not accept NDDs in the input. However for the case that $L = 0$ the BNP-DCD model was the fastest for all NLDKSS instances.

Finally we note that the solver of Glorie et al. [2014] was executed on the NLDKSS instances with a chain cap of L, for $0 \leq L \leq 4$. It was found that on average the execution time was 8.9 times slower than the fastest solver from among all the others executed on these instances as detailed at the beginning of Section 5. PICEF was the fastest solver on 40% of occasions.

5.2. Scaling experiments on realistic generated UNOS kidney exchange graphs

As motivated earlier in the paper, it is expected that kidney exchange pools will grow in size as (a) the idea of kidney exchange becomes more commonplace, and barriers to entry continue to drop, as well as (b) organized large-scale international exchanges manifest. Toward that end, in this section, we test solvers on generated compatibility graphs from a realistic simulator seeded by all historical UNOS data; the generator samples patient-donor pairs and NDDs with replacement, and draws arcs in the compatibility graph in accordance with UNOS' internal arc creation rules.

Figure 6 gives results for increasing numbers of patient-donor pairs (each column), as well as increasing numbers of non-directed donors as a percentage of the number

of patient-donor pairs (each row). As expected, as the number of patient-donor pairs increases, so too do run times for all solvers. Still, in each of the experiments, for each chain cap, both PICEF and HPIEF are on par or (typically) much faster—sometimes by orders of magnitude compared to other solvers. Appendix E in the full version of the paper gives these results in tabular form, including other statistics—minimum and maximum run times, as well as their standard deviations—that were not possible to show in Figure 6.

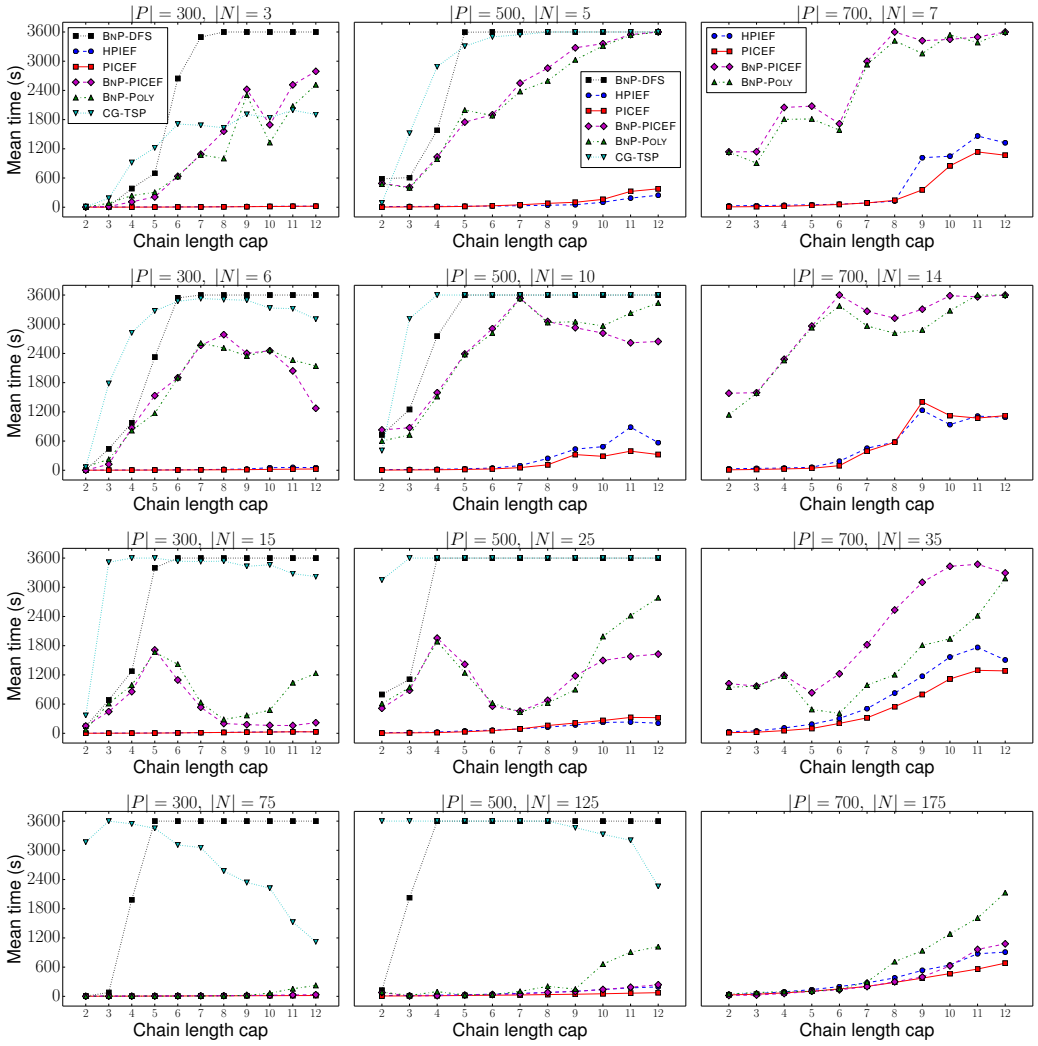

Fig. 6. Mean run time as the number of patient-donor pairs $|P| \in \{300, 500, 700\}$ increases (left to right), as the percentage of NDDs in the pool increases $|N| = \{1\%, 2\%, 5\%, 25\%\}$ of $|P|$ (top to bottom), for varying finite chain caps.

In addition to their increased scalability, we note two additional benefits of the PICEF and HPIEF models proposed in this paper: reduced variance in run time, and

relative ease of implementation when compared to other state-of-the-art solution techniques. In both the real and simulated experimental results, we find that the run time of both the PICEF and HPIEF formulations is substantially less variable than the branch-and-price-based and constraint-generation-based IP solvers. While the underlying problem being solved is NP-hard, and thus will always present worst-case instances that take substantially longer than is typical to solve, the increased predictability of the run time of these models relative to other state-of-the-art solutions—including those that are presently fielded—is attractive. Second, we note that *significant* engineering effort is involved in the creation of custom branch-and-price and constraint-generation-based codes, while both PICEF and HPIEF are implemented with relative ease, relying on only a single call to a black box IP solver.

6. FAILURE-AWARE KIDNEY EXCHANGE

Real-world exchanges all suffer to varying degrees from "last-minute" failures, where an algorithmic match or set of matches fails to move to transplantation. This can occur for a variety of reasons, including more extensive medical testing performed before a surgery, a patient or donor becoming too sick to participate, or a patient receiving an organ from another exchange or from the deceased donor waiting list.

To address these post-match arc failures, Dickerson et al. [2013] augments the standard model of kidney exchange to include a success probability p for each arc in the graph. They show how to solve this model using branch and price, where the pricing problem is solved in time exponential in the chain and cycle cap. Prior compact formulations—and, indeed, prior "edge formulations" like those due to Abraham et al. [2007], Constantino et al. [2013], and Anderson et al. [2015]—are not expressive enough to allow for generalization to this model. Intuitively, while a single arc failure prevents an entire cycle from executing, chains are capable of incremental execution, yielding utility from the NDD to the first arc failure. Thus, the expected utility gained from an arc in a chain is dependent on where in the chain that arc is located, which is not expressed in those models.

6.1. PICEF for failure-aware matching

With only minor modification, PICEF allows for implementation of failure-aware kidney exchange, under the restriction that each arc is assumed to succeed with equal probability p. While this assumption of equal probabilities is likely not true in practice, Dickerson et al. [2013] motivate why a fielded implementation of this model would potentially choose to equalize all failure probabilities: namely, so that already-sick patients—who will likely have higher failure rates—are not further marginalized by this model. Thus, given a single success probability p, we can simply rewrite the PICEF objective function as $\sum_{(i,j)\in A}\sum_{k\in\mathcal{K}'(i,j)} p^k w_{ij} y_{ijk} + \sum_{c\in\mathcal{C}} p^{|c|} w_c z_c$, which returns the maximum expected weight matching.

This objective is split into two parts: the utility received from arcs in chains, and the utility received from cycles. For the latter, a cycle c of size $|c|$ has probability $p^{|c|}$ of executing; otherwise, it yields zero utility. For the former, if an arc is used at position k in a chain, then it yields a p^k fraction of its original weight—that is, the probability that the underlying chain will execute at least through its first k arcs.

6.2. Failure-aware polynomial pricing for cycles

The initial failure-aware branch-and-price work by Dickerson et al. [2013] generalized the pricing strategy of Abraham et al. [2007], and thus suffered from a pricing problem that ran in time exponential in cycle and chain cap. Glorie et al. [2014] and Plaut et al. [2016a] discussed polynomial pricing algorithms for cycles—but not chains [Plaut et al.

Algorithm 1 Polynomial-time failure-aware pricing for cycles.

function GETDISCOUNTEDPOSITIVEPRICECYCLES($D = (V, A), K, p, w, \delta$)
  $\mathcal{C} \leftarrow \emptyset$
  for each $k = 2...K$ **do** ▷ Consider all possible cycle lengths
    $w_k(i,j) \leftarrow \delta_j - p^k w_{ij} \ \ \forall (i,j) \in A$ ▷ Reduction of Glorie et al. [2014]
    $\mathcal{C} \leftarrow \mathcal{C} \cup$ GETNEGATIVECYCLES(D, k, w_k)
  return $\mathcal{C}$

2016b]—in the *deterministic* case. Using the algorithm of Plaut et al. [2016a] as a subroutine, we present an algorithm which solves the failure-aware, or *discounted*, pricing problem for cycles in polynomial time, under the restriction that all arcs have equal success probability p.

In the deterministic setting, the price of a cycle c is $\sum_{(i,j)\in c} w_{ij} - \sum_{j \in c} \delta_j$, where w_{ij} is the weight of arc (i, j), and δ_j is the dual value of vertex j in the LP. Glorie et al. [2014] show how the arc weights and dual values can be collapsed into just arc weights, and reduce the deterministic pricing problem to finding negative-weight cycles of length at most K in a directed graph. In this section, we use "length" to denote the number of vertices in a cycle, not its weight.

The discounted price of a cycle is $p^{|c|} \sum_{(i,j)\in c} w_{ij} - \sum_{j \in c} \delta_j$. Since the utility of an arc now depends on what cycle it ends up in, we cannot collapse arc weights and dual values without knowing the length of the cycle containing it.

With this motivation, we augment the algorithm to run $O(K)$ iterations for each source vertex: one for each possible final cycle length. On each iteration, we know exactly how much arc weights will be worth in the final cycle, so we can reduce the discounted pricing problem to the deterministic pricing problem.

Pseudocode for the failure-aware cycle pricing algorithm is given by GETDISCOUNTEDPOSITIVEPRICECYCLES. Let w and δ be the arc weights and dual values respectively in the original graph. The function GETNEGATIVECYCLES is the deterministic pricing algorithm due to Plaut et al. [2016a] which returns at least one negative cycle of length at most K, or shows that none exist.

The algorithm of Plaut et al. [2016a] has complexity $O(|V||A|K^2)$. Considering all $K - 1$ possible cycle lengths brings the complexity of our algorithm to $O(|V||A|K^3)$.

THEOREM 6.1. *If there is a discounted positive price cycle in the graph, Algorithm 1 will return at least one discounted positive price cycle.*

7. CONCLUSIONS & FUTURE RESEARCH

In this paper, we addressed the tractable clearing of kidney exchanges with short cycles and long, but bounded, chains. This is motivated by kidney exchange practice, where chains are often long but bounded in length due to post-match arc failure. We introduced three IP formulations, two of which are compact, and favorably compared their LPRs to a state-of-the-art formulation with a tight relaxation. Then, on real data from the UNOS US nationwide exchange and the NLDKSS United Kingdom nationwide exchange, as well as on generated data, we showed that our new models outperform all other solvers on realistically-parameterized kidney exchange problems–often dramatically. We also explored practical extensions of our models, such as the use of branch and price for additional scalability, and an extension to the failure-aware kidney exchange case that more accurately mimics reality.

Beyond the immediate importance of more scalable static kidney exchange solvers for use in fielded exchanges, solvers like the ones presented in this paper are of practical importance in more advanced—and as yet unfielded—approaches to clearing kid-

ney exchange. In reality, patients and donors arrive to and depart from the exchange dynamically over time [Ünver 2010]. Approaches to clearing *dynamic kidney exchange* often rely on solving the static problem many times [Awasthi and Sandholm 2009; Dickerson et al. 2012; Anderson 2014; Dickerson and Sandholm 2015; Glorie et al. 2015]; thus, faster static solvers result in better dynamic exchange solutions. Use of the techniques in this paper—or adaptations thereof—as subsolvers is of interest.

From a theoretical point of view, extending the comparison of LPRs to a complete ordering of all LPRs amongst models of kidney exchange—especially for different parameterizations of the underlying model, like the inclusion of chains or arc failures—would give insight as to which solver is best suited for an exchange running under a specific set of business constraints.

ACKNOWLEDGMENTS

The authors would like to thank Ross Anderson, Kristiaan Glorie, Xenia Klimentova, Nicolau Santos, and Ana Viana for valuable discussions regarding this work and for making available their kidney exchange software for the purposes of conducting our experimental evaluation.

REFERENCES

David Abraham, Avrim Blum, and Tuomas Sandholm. 2007. Clearing Algorithms for Barter Exchange Markets: Enabling Nationwide Kidney Exchanges. In *Proc. ACM Conf. on Electronic Commerce (EC)*. 295–304.

Mohammad Akbarpour, Shengwu Li, and Shayan Oveis Gharan. 2014. Dynamic Matching Market Design. In *Proc. ACM Conf. on Economics and Computation (EC)*. 355.

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

Ross Anderson. 2014. *Stochastic models and data driven simulations for healthcare operations*. Ph.D. Dissertation. Massachusetts Institute of Technology.

Ross Anderson, Itai Ashlagi, David Gamarnik, and Alvin Roth. 2015. Finding long chains in kidney exchange using the traveling salesman problem. *Proc. National Academy of Sciences* 112, 3 (2015), 663–668.

Itai Ashlagi, Felix Fischer, Ian A Kash, and Ariel D Procaccia. 2015. Mix and Match: A strategyproof mechanism for multi-hospital kidney exchange. *Games and Economic Behavior* 91 (2015), 284–296.

Itai Ashlagi, David Gamarnik, Michael Rees, and Alvin E. Roth. 2012. The Need for (long) Chains in Kidney Exchange. NBER Working Paper No. 18202. (July 2012).

Itai Ashlagi, Patrick Jaillet, and Vahideh H. Manshadi. 2013. Kidney Exchange in Dynamic Sparse Heterogenous Pools. In *Proc. ACM Conf. on Electronic Commerce (EC)*. 25–26.

Itai Ashlagi and Alvin E Roth. 2014. Free riding and participation in large scale, multi-hospital kidney exchange. *Theoretical Economics* 9, 3 (2014), 817–863.

Pranjal Awasthi and Tuomas Sandholm. 2009. Online Stochastic Optimization in the Large: Application to Kidney Exchange. In *Proc. 21st International Joint Conf. on Artificial Intelligence (IJCAI)*. 405–411.

Cynthia Barnhart, Ellis Johnson, George Nemhauser, Martin Savelsbergh, and Pamela Vance. 1998. Branch-and-price: column generation for solving huge integer programs. *Operations Research* 46 (1998), 316–329.

Péter Biró, David F Manlove, and Romeo Rizzi. 2009. Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. *Discrete Mathematics, Algorithms and Applications* 1, 04 (2009), 499–517.

Avrim Blum, John P. Dickerson, Nika Haghtalab, Ariel D. Procaccia, Tuomas Sandholm, and Ankit Sharma. 2015. Ignorance is Almost Bliss: Near-Optimal Stochastic Matching with Few Queries. In *Proc. ACM Conf. on Economics and Computation (EC)*. 325–342.

Avrim Blum, Anupam Gupta, Ariel D. Procaccia, and Ankit Sharma. 2013. Harnessing the Power of Two Crossmatches. In *Proc. ACM Conf. on Electronic Commerce (EC)*. 123–140.

Miguel Constantino, Xenia Klimentova, Ana Viana, and Abdur Rais. 2013. New insights on integer-programming models for the kidney exchange problem. *European Journal of Operational Research* 231, 1 (2013), 57–68.

John P. Dickerson, David Manlove, Benjamin Plaut, Tuomas Sandholm, and James Trimble. 2016. Position-Indexed Formulations for Kidney Exchange. *CoRR* abs/1606.01623 (2016).

John P. Dickerson, Ariel D. Procaccia, and Tuomas Sandholm. 2012. Dynamic Matching via Weighted Myopia with Application to Kidney Exchange. In *AAAI Conf. on Artificial Intelligence (AAAI)*. 1340–1346.

John P. Dickerson, Ariel D. Procaccia, and Tuomas Sandholm. 2013. Failure-Aware Kidney Exchange. In *Proc. ACM Conf. on Electronic Commerce (EC)*. 323–340.

John P. Dickerson and Tuomas Sandholm. 2015. FutureMatch: Combining Human Value Judgments and Machine Learning to Match in Dynamic Environments. In *AAAI Conf. on Artificial Intelligence (AAAI)*. 622–628.

Yichuan Ding, Dongdong Ge, Simai He, and Christopher Ryan. 2015. A non-asymptotic approach to analyzing kidney exchange graphs. In *Proc. ACM Conf. on Economics and Computation (EC)*. 257–258.

Kristiaan Glorie, Margarida Carvalho, Miguel Constantino, Paul Bouman, and Ana Viana. 2015. Robust Models for the Kidney Exchange Problem. (2015). Working paper.

Kristiaan M. Glorie, J. Joris van de Klundert, and Albert P. M. Wagelmans. 2014. Kidney Exchange with Long Chains: An Efficient Pricing Algorithm for Clearing Barter Exchanges with Branch-and-Price. *Manufacturing & Service Operations Management (MSOM)* 16, 4 (2014), 498–512.

A. Hart, J. M. Smith, M. A. Skeans, S. K. Gustafson, D. E. Stewart, W. S. Cherikh, J. L. Wainright, G. Boyle, J. J. Snyder, B. L. Kasiske, and A. K. Israni. 2016. Kidney. *American Journal of Transplantation (Special Issue: OPTN/SRTR Annual Data Report 2014)* 16, Issue Supplement S2 (2016), 11–46.

Xenia Klimentova, Filipe Alvelos, and Ana Viana. 2014. A New Branch-and-Price Approach for the Kidney Exchange Problem. In *Computational Science and Its Applications (ICCSA-2014)*. Springer, 237–252.

Vicky Mak-Hau. 2015. On the kidney exchange problem: cardinality constrained cycle and chain problems on directed graphs: a survey of integer programming approaches. *Journal of Combinatorial Optimization* (2015), 1–25.

David Manlove and Gregg O'Malley. 2015. Paired and Altruistic Kidney Donation in the UK: Algorithms and Experimentation. *ACM Journal of Experimental Algorithmics* 19, 1 (2015).

Nicholas Mattei and Toby Walsh. 2013. Preflib: A library for preferences. In *Algorithmic Decision Theory*. Lecture Notes in Computer Science, Vol. 8176. Springer, 259–270.

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

João Pedro Pedroso. 2014. Maximizing Expectation on Vertex-Disjoint Cycle Packing. In *Computational Science and Its Applications (ICCSA-2014)*. Springer, 32–46.

Benjamin Plaut, John P. Dickerson, and Tuomas Sandholm. 2016a. Fast Optimal Clearing of Capped-Chain Barter Exchanges. In *AAAI Conf. on Artificial Intelligence (AAAI)*. 601–607.

Benjamin Plaut, John P. Dickerson, and Tuomas Sandholm. 2016b. Hardness of the Pricing Problem for Chains in Barter Exchanges. *CoRR* abs/1606.00117 (2016).

F. T. Rapaport. 1986. The case for a living emotionally related international kidney donor exchange registry. *Transplantation Proceedings* 18 (1986), 5–9.

Michael Rees, Jonathan Kopke, Ronald Pelletier, Dorry Segev, Matthew Rutter, Alfredo Fabrega, Jeffrey Rogers, Oleh Pankewycz, Janet Hiller, Alvin Roth, Tuomas Sandholm, Utku Ünver, and Robert Montgomery. 2009. A Nonsimultaneous, Extended, Altruistic-Donor Chain. *New England Journal of Medicine* 360, 11 (2009), 1096–1101.

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

Alvin Roth, Tayfun Sönmez, and Utku Ünver. 2005. Pairwise Kidney Exchange. *Journal of Economic Theory* 125, 2 (2005), 151–188.

Alvin Roth, Tayfun Sönmez, and Utku Ünver. 2007. Efficient kidney exchange: Coincidence of wants in a market with compatibility-based preferences. *American Economic Review* 97 (2007), 828–851.

Alvin Roth, Tayfun Sönmez, Utku Ünver, Frank Delmonico, and Susan L. Saidman. 2006. Utilizing list exchange and nondirected donation through 'chain' paired kidney donations. *American Journal of Transplantation* 6 (2006), 2694–2705.

Panos Toulis and David C. Parkes. 2015. Design and analysis of multi-hospital kidney exchange mechanisms using random graphs. *Games and Economic Behavior* 91, 0 (2015), 360–382.

Utku Ünver. 2010. Dynamic kidney exchange. *Review of Economic Studies* 77, 1 (2010), 372–414.

Steven Vajda. 1961. *Mathematical Programming*. Addison-Wesley.