Parameterized Algorithms for Kidney Exchange

Arnab Maiti[1] , **Palash Dey**[2]
[1,2]Indian Institute of Technology Kharagpur
[1]maitiarnab9@gmail.com, [2]palash.dey@cse.iitkgp.ac.in

Abstract

In kidney exchange programs, multiple patient-donor pairs each of whom are otherwise incompatible, exchange their donors to receive compatible kidneys. The KIDNEY EXCHANGE problem is typically modelled as a directed graph where every vertex is either an altruistic donor or a pair of patient and donor; directed edges are added from a donor to its compatible patients. The computational task is to find if there exists a collection of disjoint cycles and paths starting from altruistic donor vertices of length at most ℓ_c and ℓ_p respectively that covers at least some specific number t of non-altruistic vertices (patients). We study parameterized algorithms for the kidney exchange problem in this paper. Specifically, we design FPT algorithms parameterized by each of the following parameters: (1) the number of patients who receive kidney, (2) treewidth of the input graph + $\max\{\ell_p, \ell_c\}$, and (3) the number of vertex types in the input graph when $\ell_p \leq \ell_c$. We also present interesting algorithmic and hardness results on the kernelization complexity of the problem. Finally, we present an approximation algorithm for an important special case of KIDNEY EXCHANGE.

1 Introduction

Patients having acute renal failures are typically treated either with dialysis or with kidney transplantation. However, the quality of life on dialysis is comparitively lower and also the average life span of the patients on dialysis is around 10 years. For this reason, most patients prefer a kidney transplantation over periodic dialysis. However, the gap between the demand and supply of kidneys, which can be obtained either from a deceased person or from a living donor, is so large that the average waiting time varies from 2 to 5 years at most centers. Moreover, even if a patient is able to find a donor, there could be many medical reasons (like blood group or tissue mismatch) due to which the donor could not donate his/her kidney to the patient.

The *Kidney Paired Donation (KPD)*, a.k.a *Kidney Exchange* program, allows donors to donate their kidneys to compatible other patients with the understanding that their patients will also receive medically compatible kidneys thereby forming some kind of barter market [Alvin Roth, Tayfun Sönmez, Utku Ünver, 2004; Alvin Roth, Tayfun Sönmez, Utku Ünver, 2007; Abraham *et al.*, 2007]. Since its inception in [Rapaport, 1986], an increasing amount of people register in the kidney exchange program since, this way, patients not only have a better opportunity to receive compatible kidneys, but also can get medically better matched kidneys which last longer [Segev *et al.*, 2005]. The central problem in any kidney exchange program, also known as the *clearing problem*, is how to transplant kidneys among various patients and donors so that a maximum number of patients receive kidneys.

Typically, kidney exchange happens in a cyclical manner. The problem is represented as a directed graph: a patient along with his/her donor is a vertex; we add directed edges from a vertex u to all other vertices whose patients are compatible with the donor of the vertex u. In a (directed) cycle, the patient of every vertex receives a kidney from the donor of the previous vertex along the cycle. Donors do not have any legal obligation to donate kidneys and thus, technically speaking, a donor can leave the program as soon as his/her corresponding patient receives a kidney [Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005b; Segev *et al.*, 2005]. This is not only unfair but also leaves a patient without any donor. To avoid this problem, all kidney transplantations along a cycle are done simultaneously. As each transplantation involves two surgeries, logistic and human resource constraints allow only a few surgeries to be carried out simultaneously. For this reason, most kidney exchange platforms allow transplantation along cycles of very small length only [Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005b; Abraham *et al.*, 2007; Manlove and O'malley, 2015].

Sometimes we have *altruistic donors* (a.k.a *non-directed donors (NDDs)*) who do not have patients paired with them. This allows us to have kidney transplantations along with chains (directed paths) also starting from some altruistic vertices – an altruistic donor donates his/her kidney to some compatible patient whose paired donor donates his/her kidney to the next patient along the chain and so on. Some platform allows non-simultaneous surgeries along a chain since a broken chain is less harmful than a broken cycle — it does not leave any patient without donor [Ross Anderson and Roth, 2015]. However, broken chains also lead to unfairness and thus platforms usually only allow small chains for transplantation. Thus, in the presence of altruistic donors, the fundamental problem of kidney exchange becomes the following: find a collection of disjoint cycles and chains starting from altruistic vertices which covers a maximum number of non-altruistic vertices. We refer to Definition 1 in Section 2 for formal definition of the kidney exchange problem.

1.1 Related Work

To the best of our knowledge, [Rapaport, 1986] first introduced the idea of kidney exchange. Since then, many variants and properties have been explored [Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005b; Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005a; Segev *et al.*, 2005]. For example, a line of research allow only cycles [Constantino *et al.*, 2013; Klimentova *et al.*, 2014; Sönmez and Ünver, 2014] while others allow kidney exchange along both cycles and chains [Manlove and O'malley, 2015; Glorie *et al.*, 2014; Xiao and Wang, 2018]. All versions of the kidney exchange problem can be formulated as some suitable version of the graph packing problem. [Jia *et al.*, 2017] discussed interesting relationship between barter market and set packing.

[Krivelevich *et al.*, 2007; Abraham *et al.*, 2007] showed that the basic kidney exchange problem along with its various incarnations are NP-hard. [Krivelevich *et al.*, 2007; Jia *et al.*, 2017] developed approximation algorithms for the kidney exchange problem by exploiting interesting connection with the set packing problem. Practical heuristics and integer linear programming based algorithms haven been extensively explored for the kidney exchange problem [Manlove and O'malley, 2015; Dickerson *et al.*, 2016; Glorie *et al.*, 2014; Li *et al.*, 2014; Biro *et al.*, 2009; Klimentova *et al.*, 2014]. [Dickerson *et al.*, 2016] introduced the notion of "vertex type" and showed its usefulness as a graph parameter in real-world kidney exchange instances. Two vertices is said to have the same vertex type if their neighbourhoods are the same.

The closest predecessor of our work is by [Xiao and Wang, 2018]. They proposed an exact algorithm with running time $\mathcal{O}(2^n n^3)$ where n is the number of vertices in the underlying graph. They also presented a fixed parameter tractable algorithm for the kidney exchange problem parameterized by the number of vertex types if we do not have any restriction on the length of cycles and chains. [Lin *et al.*, 2019] studied the version of the kidney exchange problem which allows only cycles and developed a randomized parameterized algorithm with respect to the parameter being (number of patients receiving a kidney, maximum allowed length of any cycle).

1.2 Contribution

Designing exact algorithms for KIDNEY EXCHANGE has been a research focus in algorithmic game theory. We contribute to this line of research in this paper. Our specific contributions are the following.

We design FPT algorithms for the KIDNEY EXCHANGE problem parameterized by the number of patients receiving kidneys [Theorem 1], treewidth of the underlying graph + maximum length of path(ℓ_p) + maximum length of cycle allowed (ℓ_c) [Theorem 3], and the number of vertex types when $\ell_p \leq \ell_c$ [Theorem 4]. We also show that the optimization version of the KIDNEY EXCHANGE problem is a linear extended monadic second-order (EMS) extremum problem when $\max\{\ell_p, \ell_c\} = O(1)$ [Theorem 2].

We show that KIDNEY EXCHANGE admits a polynomial kernel with respect to the number of patients receiving kidneys + maximum degree when $\max\{\ell_p, \ell_c\}$ is a constant [Theorem 5]. We complement this result by showing that KIDNEY EXCHANGE does not admit any polynomial kernel parameterized by the number of patients receiving kidneys+maximum degree+$\max\{\ell_p, \ell_c\}$ unless NP $\subseteq$ co $-$ NP/poly [Theorem 6].

Finally, we also design a $(\frac{16}{9} + \epsilon)$-approximation algorithm for KIDNEY EXCHANGE if only cycles of length at most 3 are allowed (and no paths are allowed) [Corollary 1]. We believe that our work substantially improves the current theoretical understanding of the KIDNEY EXCHANGE problem.

1.3 Motivation for the Parameters

Our primary motivation for considering treewidth, $\max\{\ell_p, \ell_c\}$ and number of patients receiving kidneys as parameters is to have a faster algorithm than the existing $O(2^n)$ time algorithm from [Xiao and Wang, 2018]. Most of our parameters are typically much smaller than the graph's total number of vertices (n).

Whether a donor can donate their kidney to a patient depends on a few health parameters like blood/tissue compatibility, etc. Usually, these parameters can only take a few different values. Due to this reason, the number of vertex types is typically small. [Dickerson *et al.*, 2016] observed this phenomenon in real-world instances of kidney exchange. Hence, this parameter is useful for practical applications.

2 Preliminaries

For an integer k, we denote the sets $\{0, 1, \ldots, k\}$ and $\{1, 2, \ldots, k\}$ by $[k]_0$ and $[k]$ respectively.

A kidney exchange problem is formally represented by a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ which is known as the *compatibility graph*. A subset $\mathcal{B} \subseteq \mathcal{V}$ of vertices denotes *altruistic donors* (also called *non-directed donors*); the other set $\mathcal{V} \setminus \mathcal{B}$ of vertices denote a patient-donor pair who wish to participate in the kidney exchange program. We have a directed edge $(u, v) \in \mathcal{A}$ if the donor of the vertex $u \in \mathcal{V}$ has a kidney compatible with the patient of the vertex $v \in \mathcal{V} \setminus \mathcal{B}$. Kidney exchange happens either (i) along a *trading-cycle* $u_1, u_2, \ldots, u_k$ where the patient of the vertex $u_i \in \mathcal{V} \setminus \mathcal{B}$ receives a kidney from the donor of the vertex $u_{i-1} \in \mathcal{V} \setminus \mathcal{B}$ for every $2 \leq i \leq k$ and the patient of the vertex u_1 receives the kidney from the donor of the vertex u_k, or (ii) along a *trading-chain* $u_1, u_2, \ldots, u_k$ where $u_1 \in \mathcal{B}, u_i \in \mathcal{V} \setminus \mathcal{B}$ for $2 \leq i \leq k$ and the patient of the vertex u_j receives a kidney from the donor of the vertex u_{j-1} for $2 \leq j \leq k$. Due to operational reasons, all the kidney transplants along a trading-cycle or a trading-chain should be performed simultaneously. This puts an upper bound on the length ℓ of feasible trading-cycles and trading-chains. We define the length of a path or

$\mathcal{B}$	set of altruistic vertices
t	target number of patients to receive kidneys
ℓ_p	length of the longest path allowed
ℓ_c	length of the longest cycle allowed
τ	treewidth of underlying undirected graph
θ	number of vertex types
Δ	maximum degree of underlying undirected graph
ℓ	$\max\{\ell_p, \ell_c\}$

Table 1: Notation table.

cycle as the number of edges in it. The kidney exchange clearing problem is to find a collection of feasible trading-cycles and trading-chains which maximizes the number of patients who receive a kidney. Formally it is defined as follows.

Definition 1 (KIDNEY EXCHANGE)**.** *Given a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ with no self-loops, an altruistic vertex set $\mathcal{B} \subset \mathcal{V}$, two integers ℓ_p and ℓ_c denoting the maximum length of respectively paths and cycles allowed, and a target t, compute if there exists a collection $\mathcal{C}$ of disjoint cycles of length at most ℓ_c and paths with starting from altruistic vertices only each of length at most ℓ_p which cover at least t non-altruistic vertices. We denote an arbitrary instance of* KIDNEY EXCHANGE *by* $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$.

2.1 Graph Theoretic Terminologies

In a graph $\mathcal{G}$, $\mathcal{V}[\mathcal{G}]$ denotes the set of vertices in $\mathcal{G}$ and $\mathcal{E}[\mathcal{G}]$ denotes the set of edges in $\mathcal{G}$. $\mathcal{G}[\mathcal{V}']$ denotes the induced subgraph on $\mathcal{V}'$ where $\mathcal{V}' \subseteq \mathcal{V}[\mathcal{G}]$. Two vertices u and v in a directed graph $\mathcal{G}$ are called vertices of the same type if they have the same set of in-neighbors and the same set of out-neighbors. If there are no self loops in $\mathcal{G}$, vertices of the same type form an independent set. Treewidth measures how treelike an undirected graph is. We refer to [Cygan *et al.*, 2015] for an elaborate description of treewidth, tree decomposition, and nice tree decomposition. Since our graph is directed, whenever we mention treewidth of our graph, we refer to the treewidth of the underlying undirected graph; two vertices u and v are neighbors of the underlying undirected graph if and only if either there is an edge from u to v or from v to u. Also refer to the Table 1 for the important notations.

2.2 Standard Definitions

Due to space constraints we refer the reader to the full version of our paper [Maiti and Dey, 2021] for the formal introduction of Parameterized Complexity and for the definitions of kernalization, cross composition, X3C$'$ problem and 3-SET PACKING problem. For a detailed discussion on extended monadic second-order extremum problems, please refer to [Arnborg *et al.*, 1991].

3 Results

We present our results in this section. Due to space constraints, we have omitted few proofs. They are marked by ($\star$) and they are available in the full version of our paper [Maiti and Dey, 2021].

We begin with presenting an FPT algorithm for the KIDNEY EXCHANGE problem parameterized by the number of patients who receive a kidney. We use the technique of color coding [Alon *et al.*, 1995] to design our algorithm.

Theorem 1. *There is a algorithm for the* KIDNEY EXCHANGE *problem which runs in time $\mathcal{O}^*(2^{\mathcal{O}(t)})$.*

Proof. Let $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$ be an arbitrary instance of KIDNEY EXCHANGE. If $\ell_p \geq t$, we check whether there exists a path starting from an altruistic vertex of length t; if $\ell_c \geq t$, then we check whether there exists a cycle of length ℓ_1 for some $t \leq \ell_1 \leq \ell_c$. Note that this can be checked by a deterministic algorithm in time $\mathcal{O}^*(2^{\mathcal{O}(t)})$ (for path see [Cygan *et al.*, 2015], for cycle see [Zehavi, 2016]). If there exists such a cycle or path then clearly $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$ is a YES instance. So for the rest of the proof , let us update l_p and l_c as $\min\{\ell_p, t-1\}$ and $\min\{\ell_c, t-1\}$ respectively.

We observe that if $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$ is a YES instance, then there is a collection $\mathcal{C} = (\mathcal{D}_1, \ldots, \mathcal{D}_k)$ of $k \leq t$ disjoint cycles and paths starting from altruistic vertices each of length at most ℓ_c and ℓ_p respectively which covers t_1 non-altruistic vertices where $t \leq t_1 \leq 2t$. Note that such a collection will exist as $\ell_c < t$ and $\ell_p < t$. We observe that the total number of vertices involved in $\mathcal{D}_1, \ldots, \mathcal{D}_k$ is at most $3t$ (at most one altruistic vertex in each $\mathcal{D}_i, i \in [k]$). We color each vertex of $\mathcal{G}$ uniformly at random from a set $\mathcal{S}$ of $3t$ colors. We say a coloring of $\mathcal{G}$ is good if every vertex in the $\mathcal{C}$ gets a different color. A random coloring is good with probability at least

$$\frac{(3t)!}{(3t)^{3t}} \geq \frac{1}{e^{3t}}$$

Let C be a non-empty set of colors. Let $\mathcal{G}_C$ denote an induced subgraph of $\mathcal{G}$ on the set of vertices colored with one of the colors in C. We now solve the KIDNEY EXCHANGE problem using dynamic programming. We maintain a dynamic programming table $\mathcal{D}$ which is indexed by a set of colors C and a number t'. $\mathcal{D}(C, t')$ denotes whether there is a collection of disjoint cycles and paths starting from altruistic vertices each of length at most ℓ_c and ℓ_p respectively which covers t' non-altruistic vertices in the graph $\mathcal{G}_C$ and no two vertices in this collection have the same color. Now we introduce a function f which has a set of colors C' and a number i as its arguments. $f(C', i)$ decides whether there is a valid colourful cycle or path starting from altruistic vertex of length i in $\mathcal{G}_{C'}$. By valid colourful cycle (resp. path) we mean that no two vertices in the cycle (resp. path) have the same color and the length of the cycle (resp. path) is at most ℓ_c (resp. ℓ_p). $f(C', i)$ can be computed in time $\mathcal{O}^*(2^{\mathcal{O}(i)})$ time (see [Cygan *et al.*, 2015]). Now we present a recursive formula to compute each entry of the table.

$$\mathcal{D}(C, t') = \bigvee_{C', i : \emptyset \neq C' \subseteq C, i \in [t']} (\mathcal{D}(C \setminus C', t' - i) \wedge f(C', i))$$

For base cases, let $\mathcal{D}(C, 0) = 1$ and $\mathcal{D}(\emptyset, t') = 0$ if $t' > 0$. Now argue the correctness of the above recursive equation. In one direction, let $\mathcal{D}(C, t') = 1$. It implies that there is a collection of valid colourful disjoint cycles and paths starting from altruistic vertices which covers t' non-altruistic vertices in the graph $\mathcal{G}_C$. Now consider once such disjoint cycle or path. Let the number of non-atruistic vertices in it be i and the set of colors of the vertices in it be C'. Then clearly $f(C', i) = 1$ and $\mathcal{D}(C \setminus C', t - i) = 1$. In the other direction, let there exist a set $C' \subseteq C$ and a number $i \in [t']$ such that $\mathcal{D}(C \setminus C', t' - i) \wedge f(C', i) = 1$. This implies that there is a collection of valid colourful disjoint cycles and paths starting from altruistic vertices which covers $t' - i$ non-altruistic vertices in the graph $\mathcal{G}_{C \setminus C'}$ and there is a valid colourful cycle or path starting from altruistic vertex of length i in $\mathcal{G}_{C'}$. Hence there is a collection of valid colourful disjoint cycles and paths starting from altruistic vertices which covers t' non-altruistic vertices in the graph $\mathcal{G}_C$. Therefore $\mathcal{D}(C, t') = 1$.

If $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$ is a YES instance and coloring is good then $\bigvee_{t \leq t' \leq 2t} \mathcal{D}(\mathcal{S}, t') = 1$. For a NO instance, $\bigvee_{t \leq t' \leq 2t} \mathcal{D}(\mathcal{S}, t') = 0$ for any coloring.

The total number of entries in the table $\mathcal{D}$ is $2^{O(t)} \cdot t$. Each entry can be computed in time $\mathcal{O}^*(2^{\mathcal{O}(\ell)})$. Hence, with probability at least e^{-3t}, our algorithm outputs the correct decision in $\mathcal{O}^*(2^{\mathcal{O}(t)})$ time. By repeating $\mathcal{O}(e^{3t})$ times, we find the correct decision with constant probability. The overall running time of our algorithm is $\mathcal{O}^*(2^{\mathcal{O}(t)})$. The algorithm can be derandomized by using a $(n, 3t, 3t)$-splitter. ☐

We now consider the parameter treewidth to design an FPT algorithm. Towards that, we first show the following.

Theorem 2 ($\star$). *The optimization version of the* KIDNEY EXCHANGE *problem is a linear extended monadic second-order (EMS) extremum problem when* $\max\{\ell_p, \ell_c\} = O(1)$.

It follows immediately from Theorem 2 and [Arnborg *et al.*, 1991] that KIDNEY EXCHANGE is FPT parameterized by τ (treewidth) and $\ell = \max\{\ell_p, \ell_c\}$. However, the running time that we get is not practically useful. Next we proceed to design an efficient dynamic programming based algorithm with running time $\mathcal{O}^*(\ell^{O(\tau)} \tau^{O(\tau)})$.

Theorem 3 ($\star$). *There is an algorithm for the* KIDNEY EXCHANGE *problem which runs in time* $\mathcal{O}^*(\ell^{O(\tau)} \tau^{O(\tau)})$.

Proof Sketch. Let $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t')$ be an arbitrary instance of KIDNEY EXCHANGE. Let us assume for the rest of proof that $\ell_c = \ell_p$ for the sake of simplicity. The proof can be easily extended to the case where $\ell_c \neq \ell_p$. Let $\mathcal{T} = (T, \{X_t \subseteq \mathcal{V}[\mathcal{G}]\}_{t \in V(T)})$ be a nice tree decomposition of the input n-vertex graph $\mathcal{G}$ that has width at most τ and has **introduce edge nodes**. Let $\mathcal{T}$ be rooted at some node r. For a node t of $\mathcal{T}$, let V_t be the union of all the bags present in the subtree of $\mathcal{T}$ rooted at t, including X_t. Let $\mathcal{G}_t = (V_t, E_t = \{e : e$ is introduced in the subtree rooted at $t\})$ be a subgraph of $\mathcal{G}[V_t]$. We solve the KIDNEY EXCHANGE problem using dynamic programming. At every bag of the tree decomposition, we maintain a dynamic programming table $\mathcal{D}$ which is indexed by the tuple $(u, (P_i)_{i \in [\mu]_0}, (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]}, g : [\mu] \to \{p, c\}, \mathcal{L} : [\mu] \to [\ell]_0, a : [\mu] \to \{0, 1\}, s : [\mu] \to X_u \cup \{\perp\}, e : [\mu] \to X_u \cup \{\perp\})$ where u is the node in the tree decomposition, $\mu \in [\tau + 1]$, $(P_i)_{i \in [\mu]_0}$ is a partition of X_u, Q_i is a permutation of P_i, and E_i is a set of edges from $\mathcal{E}[\mathcal{G}_u]$ having both their end points in P_i for every $i \in [\mu]$. Here $\perp$ is a special symbol which is not part of $\mathcal{V}[\mathcal{G}]$. We define $\mathcal{D}(u, (P_i)_{i \in [\mu]_0}, (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]}, g, \mathcal{L}, a, s, e)$ to be the maximum number of edges present in any subgraph $\mathcal{H}$ of $\mathcal{G}_u$ such that all the following conditions hold.

- $\mathcal{H}$ excluding the isolated vertices is a collection of disjoint paths and disjoint cycles of length at most ℓ.
- Disjoint paths which do not have a vertex from X_u must begin with an altruistic vertex.
- $\mathcal{V}[\mathcal{H}] \cap X_u = X_u \setminus P_0$
- For each path $P \in \mathcal{H}$, if $\exists i, j \in [\mu]$ such that $P_i \cap \mathcal{V}[P] \neq \emptyset$ and $P_j \cap \mathcal{V}[P] \neq \emptyset$ then $i = j$.
- For each cycle $C \in \mathcal{H}$, if $\exists i, j \in [\mu]$ such that $P_i \cap \mathcal{V}[C] \neq \emptyset$ and $P_j \cap \mathcal{V}[C] \neq \emptyset$ then $i = j$.
- For every $i \in [\mu]$, if $g(i) = p$, then either P_i consists of a only one vertex which is isolated or $\mathcal{E}[\mathcal{H}] \cap \mathcal{E}[\mathcal{G}_u[P_i]] \subseteq \mathcal{E}[P]$ where P is a disjoint path in $\mathcal{H}$, Q_i is a topological order of P_i w.r.t. P, $E_i = \mathcal{E}[\mathcal{H}] \cap \mathcal{E}[\mathcal{G}_u[P_i]]$, P has $\mathcal{L}(i)$ edges and exactly $a(i)$ altruistic vertex. If $s(i) = \perp$ then starting vertex of P is not part of P_i. If $s(i) = v$ then starting vertex of P is v where $v \in P_i$. If $e(i) = \perp$ then ending vertex of P is not part of P_i. If $e(i) = v$ then ending vertex of P is v where $v \in P_i$.
- For every $i \in [\mu]$, if $g(i) = c$ and $Q_i = x_1 > \cdots > x_\nu$, then $\mathcal{E}[\mathcal{H}] \cap \mathcal{E}[\mathcal{G}_u[P_i]] \subseteq \mathcal{E}[C]$ where C is a disjoint cycle in $\mathcal{H}$, Q_i is a topological order of P_i w.r.t. the path created by removing the edge (x_ν, x_1) from C, $E_i = \mathcal{E}[\mathcal{H}] \cap \mathcal{E}[\mathcal{G}_u[P_i]]$, C has $\mathcal{L}(i)$ edges.

Clearly $\mathcal{D}(r, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset) \geq t$ iff $(\mathcal{G}, \mathcal{B}, \ell_p, \ell_c, t)$ is a YES instance.

We now explain how we update the table entry $\mathcal{D}(u, (P_i)_{i \in [\mu]_0}, (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]}, g, \mathcal{L}, a, s, e)$. First, we make the table entry $-\infty$ if we encounter a DP table index which can't lead to a feasible solution. We now use the following formulas to compute table entries in a bottom-up fashion.

Leaf node: Since the tree-decomposition $(\mathcal{T}, \{X_u\}_{u \in \mathcal{V}[\mathcal{T}]})$ is nice, for every leaf node u of $\mathcal{T}$, we have $X_u = \emptyset$ and thus all the DP table entries at u is set to 0.

Introduce vertex node: Let $u \in \mathcal{T}$ be an introduce-vertex-node with child $u' \in \mathcal{T}$ such that $X_u = X_{u'} \cup \{w\}$ for some vertex $w \in \mathcal{V}[\mathcal{G}] \setminus X_{u'}$. Since no edge is introduced in an introduce-vertex-node, w is an isolated vertex in $\mathcal{G}_u$. We set the table entry $\mathcal{D}(u, (P_i)_{i \in [\mu]_0}, (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]} g, \mathcal{L}, a, s, e)$ to $\mathcal{D}(u', (P_j \setminus \{w\}, (P_i)_{i \in [\mu]_0 \setminus \{j\}}), (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]} g, \mathcal{L}, a, s, e)$ if $w \in P_j$ where $j = 0$ or $P_j = \{w\}$, $g(j) = p$, $\mathcal{L}(j) = 0$, $s(j) = \{w\}$, $e(j) = \{w\}$, $a(j) = \mathbb{1}(w \in \mathcal{B})$ where $\mathbb{1}(\cdot)$ is the indicator function otherwise we set it to $-\infty$.

Introduce edge node: Let $u \in \mathcal{T}$ be an introduce-edge-node introducing an edge $(x, y) \in \mathcal{E}[\mathcal{G}]$ and u' the child of u. That is, we have $X_u = X_{u'}$. For the table entry $\mathcal{D}(u, (P_i)_{i \in [\mu]_0}, (Q_i)_{i \in [\mu]}, (E_i)_{i \in [\mu]}, g, \mathcal{L}, a, s, e)$, if any of the following conditions hold

- There does not exist any $i \in [\mu]$ such that $x, y \in P_i$
- There exists an $i \in [\mu]$ such that $x, y \in P_i$ and $(x, y) \notin E_i$

then we set the entry to $\mathcal{D}(u', (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)$. Otherwise, let us assume that there exists a $j \in [\mu]$ such that $x, y \in P_j$ and $(x, y) \in E_j$. Intuitively, we compute the appropriate DP table indices at node u' that we can get when we remove the edge $\{x, y\}$ from the current DP table index at node u. Then we choose the maximum value among the DP table entries corresponding to DP table indices at node u' that we computed and add 1 to it to get the value for the current table entry.

Forget vertex node: Let $u \in \mathcal{T}$ be a forget node with a child u' such that $X_u = X_{u'} \setminus \{w\}$ for some $w \in X_{u'}$. In this case, we update the table entry $\mathcal{D}(u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)$ as follows. For an index of the DP table at node u', we define a function DEL(.). Intuively, applying DEL(.) on a DP table index at node u' leads to a DP table index at node u that we get when we remove w from the DP table index at node u'.

We update the table entry $\mathcal{D}(u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)$ as

$$\max\{\mathcal{D}(u', (P''_i)_{i\in[\mu]_0}, (Q''_i)_{i\in[\mu]}, (E''_i)_{i\in[\mu]}, g'', \mathcal{L}'', a'', s'', e'') : \\ \text{DEL}(u', (P''_i)_{i\in[\mu]_0}, (Q''_i)_{i\in[\mu]}, (E''_i)_{i\in[\mu]}, g'', \mathcal{L}'', a'', s'', e'') \\ = (u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)\}$$

Join node: For a join node u, let u_1, u_2 be its two children. Note that $X_u = X_{u_1} = X_{u_2}$. Now we introduce the notion of compatibility. Inituitively, we say that the following pair of tuple $((u_1, (P^1_i)_{i\in[\mu_1]_0}, (Q^1_i)_{i\in[\mu_1]}, (E^1_i)_{i\in[\mu_1]}, g^1, \mathcal{L}^1, a^1, s^1, e^1),$ $(u_2, (P^2_i)_{i\in[\mu_2]_0}, (Q^2_i)_{i\in[\mu_2]}, (E^2_i)_{i\in[\mu_2]}, g^2, \mathcal{L}^2, a^2, s^2, e^2))$ is said to be compatible with the current DP table index $(u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)$ if partitions in the pair of tuples can be used to construct the current DP table index at node u. For instance, we check whether we can use the paths corresponding to the partitions in the pair to construct the paths corresponding to the partition in the current DP table index at node u.

Let T^C be the set of all pair of tuples (T_1, T_2) which are compatible as per the above conditions with respect to $(u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e)$. Then

$$\mathcal{D}(u, (P_i)_{i\in[\mu]_0}, (Q_i)_{i\in[\mu]}, (E_i)_{i\in[\mu]}, g, \mathcal{L}, a, s, e) \\ = \max_{(T_1,T_2)\in T^C} \mathcal{D}(T_1) + \mathcal{D}(T_2)$$

Each table entry can be updated in $\ell^{\mathcal{O}(\tau)}\tau^{\mathcal{O}(\tau)}$ time and the size of each table in any node of the tree decomposition is $\ell^{\mathcal{O}(\tau)}\tau^{\mathcal{O}(\tau)}$. Hence, the running time of our algorithm is $\mathcal{O}^*(\ell^{\mathcal{O}(\tau)}\tau^{\mathcal{O}(\tau)})$. □

Let θ denote the number of vertex types in a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. [Xiao and Wang, 2018] presented an FPT algorithm parameterized by θ when $\ell_p = \ell_c = |\mathcal{V}|$. We now improve the result by presenting our FPT algorithm parameterized by θ when $\ell_p \leq \ell_c$. Towards that, we first present an important lemma on the structure of an optimal solution.

Lemma 1 (⋆)**.** *In every* KIDNEY EXCHANGE *problem instance when $\ell_p \leq \ell_c$, there exists an optimal solution where the length of every path and cycle in that solution is at most $\theta + 3$.*

We use the following useful result by Lenstra to design our FPT algorithm.

Lemma 2 (Lenstra's Theorem [Lenstra Jr, 1983])**.** *There is an algorithm for computing a feasible as well as an optimal solution of an integer linear program which is fixed parameter tractable parameterized by the number of variables.*

Theorem 4. *There exists an* FPT *algorithm for the* KIDNEY EXCHANGE *problem parameterized by θ when $\ell_p \leq \ell_c$.*

Proof. Let $(\mathcal{G} = (\mathcal{V}, \mathcal{E}), \mathcal{B}, \ell_p, \ell_c, t)$ be an arbitrary instance of KIDNEY EXCHANGE where $\ell_p \leq \ell_c$. We denote the set of types in $\mathcal{G}$ by Γ and the type of vertex v by $\gamma(v)$. For a type $\gamma \in \Gamma$, let the set of vertices of type γ be $V_\gamma \subseteq \mathcal{V}$. If there is a $\gamma \in \Gamma$ such that V_γ contains both altrusitic and non-altruistic vertices, then we remove the non-altruistic vertices as they have in degree 0 and can't be part of any feasible solution. For a path/cycle $p = v_1 \ldots v_k$, "signature of p" is defined as $\gamma(v_1) \ldots \gamma(v_k)$ and we denote it by $\gamma(p)$. For a type $\gamma \in \Gamma$, we denote the number of vertices of type γ in $\mathcal{G}$ by $n(\gamma)$. For a type $\gamma \in \Gamma$ and a path/cycle/signature-sequence p, we denote the number of vertices of type γ in p by $n_p(\gamma)$. For a path/cycle/signature-sequence p, we denote the number of non-altruistic vertices in p by $\lambda(p)$. Let $\mathcal{A}$ denote the set of signatures of paths starting from altruistic vertices of length at most $\min\{\theta + 4, \ell_p\}$ and signatures of cycles of length at most $\min\{\theta + 3, \ell_c\}$ in $\mathcal{G}$. Since there are θ types, we have $|\mathcal{A}| = \mathcal{O}(\theta^\theta)$. We can compute the set $\mathcal{A}$ in time $\mathcal{O}^*(\theta^\theta)$ as follows. For each possible signature-sequence $p = \gamma_1 \ldots \gamma_k$ of a path, we check if there is an edge from a vertex in V_{γ_i} to a vertex in $V_{\gamma_{i+1}}$ for all $i \in [k-1]$ and $n_p(\gamma) \leq n(\gamma)$ for all $\gamma \in \Gamma$. If the conditions hold true, then we add p to the set $\mathcal{A}$. Similarly for each possible signature-sequence $p = \gamma_1 \ldots \gamma_k$ of a path, we check if there is an edge from a vertex in V_{γ_i} to a vertex in $V_{\gamma_{(i \bmod k)+1}}$ for all $i \in [k]$ and $n_p(\gamma) \leq n(\gamma)$ for all $\gamma \in \Gamma$. If the conditions hold true, then we add p to the set $\mathcal{A}$.

We consider the following integer linear program; its variables are $x(p)$ for every $p \in \mathcal{A}$.

$$\max \sum_{p\in\mathcal{A}} \lambda(p)x(p)$$

$$\text{subject to:} \sum_{p\in\mathcal{A}} n_p(\gamma)x(p) \leq n(\gamma) \qquad \forall\gamma \in \Gamma$$

$$x(p) \in \{0, 1, \ldots, n\} \qquad \forall p \in \mathcal{A}$$

We claim that the KIDNEY EXCHANGE instance is a YES instance if and only if the optimal value of the above ILP is at least t.

In one direction, suppose the KIDNEY EXCHANGE instance is a YES instance. Let $\mathcal{C}$ be an optimal solution (a multi-set of paths and cycles) of the KIDNEY EXCHANGE instance. For signature sequence $p \in \mathcal{A}$, we define $x(p) = \sum_{q \in \mathcal{C}} \mathbb{1}(p = \gamma(q))$ where $\mathbb{1}(\cdot)$ is the indicator function. As $\mathcal{C}$ is a subgraph of $\mathcal{G}$, we have $\sum_{p \in \mathcal{A}} n_p(\gamma) \sum_{q \in \mathcal{C}} \mathbb{1}(p = \gamma(q)) \leq n(\gamma), \forall \gamma \in \Gamma$. Hence $x(p)_{p \in \mathcal{A}}$ is a feasible solution. Since $\mathcal{C}$ covers at least t non-altruistic vertices, it follows that $\sum_{p \in \mathcal{A}} \lambda(p)x(p) \geq t$. Hence the optimal value of the above ILP is at least t.

On the other direction, suppose there exists a solution $(x^*(p))_{p \in \mathcal{A}}$ to the ILP such that $\sum_{p \in \mathcal{A}} \lambda(p)x^*(p) \geq t$. We now describe an iterative approach to construct a solution $\mathcal{C}$ for the KIDNEY EXCHANGE instance from $(x^*(p))_{p \in \mathcal{A}}$. We initialize $\mathcal{C}$ to the empty set and initialize $x'(w)$ to 0 for all $w \in \mathcal{A}$. Let $n_{\mathcal{C}}(\gamma)$ denote the number of vertices of type γ in $\mathcal{C}$. Till there exists a $w \in \mathcal{A}$ such that $x'(w) < x^*(w)$, we add to $\mathcal{C}$ a path/cycle q of signature w belonging to $\mathcal{G} \setminus \mathcal{C}$ and increase $x'(w)$ by 1. Now we show that if there exists a $w \in \mathcal{A}$ such that $x'(w) < x^*(w)$, then there is always a path/cycle q of signature w in $\mathcal{G} \setminus \mathcal{C}$. Due to the way $\mathcal{A}$ is defined and the fact that $w \in \mathcal{A}$, it suffices to show that $n_w(\gamma) \leq n(\gamma) - n_{\mathcal{C}}(\gamma), \forall \gamma \in \Gamma$. Since $(x^*(p))_{p \in \mathcal{A}}$ is a feasible solution, $\sum_{p \in \mathcal{A}} n_p(\gamma)x^*(p) \leq n(\gamma), \forall \gamma \in \Gamma$. Therefore $n_w(\gamma) \leq \sum_{p \in \mathcal{A}} n_p(\gamma)(x^*(p) - x'(p)) \leq n(\gamma) - \sum_{p \in \mathcal{A}} n_p(\gamma)x'(w) = n(\gamma) - n_{\mathcal{C}}(\gamma), \forall \gamma \in \Gamma$. Hence there is always a path/cycle q of signature w. Now when the iterative procedure terminates, the number of non-altruistic vertices covered by $\mathcal{C}$ is $\sum_{p \in \mathcal{A}} \lambda(p)x^*(p)$ which is at least t. Hence, the KIDNEY EXCHANGE instance is a YES instance. Now the result follows from Lemma 2. □

We now present our parameterized hardness result. Our parameter is $\Delta + \ell_p + \ell_c$. It turns out that the proof of [Abraham *et al.*, 2007, Theorem 1] which shows NP-completeness of the kidney exchange problem can be appropriately modified to get Observation 1. We use X3C′ problem for that which is known to be NP-complete [Gonzalez, 1985].

Observation 1 ($\star$)**.** *The* KIDNEY EXCHANGE *problem, parameterized by $\Delta + \ell_p + \ell_c$, is para-*NP*-hard.*

We now present our result on kernelization. We show that the KIDNEY EXCHANGE problem admits a polynomial kernel for the parameter $t + \Delta$ for every constant ℓ_p and ℓ_c.

Theorem 5 ($\star$)**.** *For the* KIDNEY EXCHANGE *problem, there exists a vertex kernel of size $\mathcal{O}(t\Delta^{\max\{\ell_p, \ell_c\}})$.*

Theorem 5 raises the following question: does the KIDNEY EXCHANGE problem admit a polynomial kernel parameterized by $t + \Delta + \ell_p + \ell_c$? We answer this question negatively in Theorem 6 using cross composition.

Theorem 6 ($\star$)**.** KIDNEY EXCHANGE *does not admit any polynomial kernel with respect to the parameter $t+\Delta+\ell_p+\ell_c$ unless* $\mathsf{NP} \subseteq \mathsf{co-NP}$*/poly.*

We now present our approximation result for the KIDNEY EXCHANGE problem when only cycles of length at most 3 are allowed; no path is allowed. [Biro *et al.*, 2009] studied this problem with the name MAX SIZE ≤3-WAY EXCHANGE and proved APX-hardness. A trivial extension from the result on MAX CYCLE WEIGHT≤k-WAY EXCHANGE in [Biro *et al.*, 2009] leads to a 2+ε approximation algorithm for MAX SIZE ≤3-WAY EXCHANGE. Now towards designing the approximation algorithm, we use the following result for 3-SET PACKING problem which is due to [Cygan, 2013].

Lemma 3. *For every $\varepsilon > 0$, there is $(4/3+\varepsilon)$-approximation algorithm for the* 3-SET PACKING *problem for optimizing k.*

The following result relates MAX SIZE ≤3-WAY EXCHANGE to 3-SET PACKING.

Theorem 7 ($\star$)**.** *If there is a α-approximation algorithm for* 3-SET PACKING *problem, then there is a $\frac{4\alpha}{3}$-approximation algorithm for* MAX SIZE ≤3-WAY EXCHANGE.

Theorem 7 immediately gives us the following corollary.

Corollary 1. *For every $\varepsilon > 0$, there is a $(\frac{16}{9} + \varepsilon)$-approximation algorithm for* KIDNEY EXCHANGE *if only cycles of length at most* 3 *are allowed (and no paths are allowed).*

4 Conclusion and Open Questions

In this paper, we have presented FPT algorithms for the KIDNEY EXCHANGE problem with respect to some natural parameters, namely (i) solution size t (the number of patients receiving a kidney), (ii) treewidth+$\max\{\ell_p, \ell_c\}$ and (iii) the number of vertex types when $\ell_p \leq \ell_c$. For kernelization, we have exhibited a polynomial kernel w.r.t $\Delta + t$ when $\max\{\ell_p, \ell_c\}$ is $\mathcal{O}(1)$. We have complemented this result by refuting existence of a polynomial kernel parameterized $\Delta + t + \max\{\ell_p, \ell_c\}$ unless $\mathsf{NP} \subseteq \mathsf{co-NP/poly}$. We have finally presented an $(\frac{16}{9} + \epsilon)$-approximation algorithm for KIDNEY EXCHANGE in the special case when cycles of length at most 3 are allowed and no path is allowed.

Our work leaves many interesting open questions. One such question is whether KIDNEY EXCHANGE is FPT w.r.t treewidth only. Another such question is whether KIDNEY EXCHANGE is FPT w.r.t number of vertex types (without any assumption on ℓ_c and ℓ_p). Another important question is the existence of a polynomial kernel for KIDNEY EXCHANGE parameterized by the solution size alone when the maximum allowed length of paths and cycles are the same. Our hardness proof in Theorem 6 breaks down if we want to allow paths and cycles of the same length. Finally, it is interesting to study whether the running time of our FPT algorithms can be improved further or are they best possible assuming standard complexity-theoretic assumptions like ETH or SETH.

References

[Abraham *et al.*, 2007] David J. Abraham, Avrim Blum, and Tuomas Sandholm. Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. In Jeffrey K. MacKie-Mason, David C. Parkes, and Paul Resnick, editors, *Proc. 8th ACM Conference on Electronic Commerce (EC-2007)*, pages 295–304. ACM, 2007.

[Alon *et al.*, 1995] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM (JACM)*, 42(4):844–856, 1995.

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

[Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005a] Alvin Roth, Tayfun Sönmez, Utku Ünver. A kidney exchange clearinghouse in new england. *Am. Econ. Rev*, 95(2):376–380, 2005.

[Alvin Roth, Tayfun Sönmez, Utku Ünver, 2005b] Alvin Roth, Tayfun Sönmez, Utku Ünver. Pairwise kidney exchange. *J. Econ. Theory*, 125(2):151–188, 2005.

[Alvin Roth, Tayfun Sönmez, Utku Ünver, 2007] Alvin Roth, Tayfun Sönmez, Utku Ünver. Efficient kidney exchange: Coincidence of wants in markets with compatibility-based preferences. *Am. Econ. Rev*, pages 828–851, 2007.

[Arnborg *et al.*, 1991] Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, 1991.

[Biro *et al.*, 2009] Péter Biro, David F Manlove, and Romeo Rizzi. Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. *Discrete Math Algorithms Appl*, 1(04):499–517, 2009.

[Constantino *et al.*, 2013] Miguel Constantino, Xenia Klimentova, Ana Viana, and Abdur Rais. New insights on integer-programming models for the kidney exchange problem. *Eur. J. Oper. Res.*, 231(1):57–68, 2013.

[Cygan *et al.*, 2015] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

[Cygan, 2013] Marek Cygan. Improved approximation for 3-dimensional matching via bounded pathwidth local search. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 509–518. IEEE, 2013.

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

[Glorie *et al.*, 2014] Kristiaan M Glorie, J Joris van de Klundert, and Albert PM Wagelmans. Kidney exchange with long chains: An efficient pricing algorithm for clearing barter exchanges with branch-and-price. *Manuf. Serv. Oper. Manag.*, 16(4):498–512, 2014.

[Gonzalez, 1985] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.

[Jia *et al.*, 2017] Zhipeng Jia, Pingzhong Tang, Ruosong Wang, and Hanrui Zhang. Efficient near-optimal algorithms for barter exchange. In *Proc. 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 362–370, 2017.

[Klimentova *et al.*, 2014] Xenia Klimentova, Filipe Alvelos, and Ana Viana. A new branch-and-price approach for the kidney exchange problem. In *Proc. International Conference on Computational Science and Its Applications*, pages 237–252. Springer, 2014.

[Krivelevich *et al.*, 2007] Michael Krivelevich, Zeev Nutov, Mohammad R Salavatipour, Jacques Verstraete, and Raphael Yuster. Approximation algorithms and hardness results for cycle packing problems. *ACM T Algorithms*, 3(4):48–es, 2007.

[Lenstra Jr, 1983] Hendrik W Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of operations research*, 8(4):538–548, 1983.

[Li *et al.*, 2014] Jian Li, Yicheng Liu, Lingxiao Huang, and Pingzhong Tang. Egalitarian pairwise kidney exchange: fast algorithms via linear programming and parametric flow. In *AAMAS*, pages 445–452, 2014.

[Lin *et al.*, 2019] Mugang Lin, Jianxin Wang, Qilong Feng, and Bin Fu. Randomized parameterized algorithms for the kidney exchange problem. *Algorithms*, 12(2):50, 2019.

[Maiti and Dey, 2021] Arnab Maiti and Palash Dey. Parameterized algorithms for kidney exchange. *arXiv preprint arXiv:2112.10250*, 2021.

[Manlove and O'malley, 2015] David F Manlove and Gregg O'malley. Paired and altruistic kidney donation in the uk: Algorithms and experimentation. *J. Exp. Algorithmics*, 19:1–21, 2015.

[Rapaport, 1986] Felix T Rapaport. The case for a living emotionally related international kidney donor exchange registry. In *Transplantation proceedings*, volume 18, page 5, 1986.

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

[Segev *et al.*, 2005] Dorry L Segev, Sommer E Gentry, Daniel S Warren, Brigitte Reeb, and Robert A Montgomery. Kidney paired donation and optimizing the use of live donor organs. *Jama*, 293(15):1883–1890, 2005.

[Sönmez and Ünver, 2014] Tayfun Sönmez and M Utku Ünver. Altruistically unbalanced kidney exchange. *J. Econ. Theory*, 152:105–129, 2014.

[Xiao and Wang, 2018] Mingyu Xiao and Xuanbei Wang. Exact algorithms and complexity of kidney exchange. In *IJCAI*, pages 555–561, 2018.

[Zehavi, 2016] Meirav Zehavi. A randomized algorithm for long directed cycle. *Information Processing Letters*, 116(6):419–422, 2016.