J Comb Optim (2017) 33:35–59
DOI 10.1007/s10878-015-9932-4

On the kidney exchange problem: cardinality constrained cycle and chain problems on directed graphs: a survey of integer programming approaches

Vicky Mak-Hau[1]

Published online: 17 July 2015

Abstract The Kidney Exchange Problem (KEP) is a combinatorial optimization problem and has attracted the attention from the community of integer programming/combinatorial optimisation in the past few years. Defined on a directed graph, the KEP has two variations: one concerns cycles only, and the other, cycles as well as chains on the same graph. We call the former a Cardinality Constrained Multi-cycle Problem (CCMcP) and the latter a Cardinality Constrained Cycles and Chains Problem (CCCCP). The cardinality for cycles is restricted in both CCMcP and CCCCP. As for chains, some studies in the literature considered cardinality restrictions, whereas others did not. The CCMcP can be viewed as an Asymmetric Travelling Salesman Problem that *does* allow subtours, however these subtours are constrained by cardinality, and that it is not necessary to visit all vertices. In existing literature of the KEP, the cardinality constraint for cycles is usually considered to be small (to the best of our knowledge, no more than six). In a CCCCP, each vertex on the directed graph can be included in at most one cycle or chain, but not both. The CCMcP and the CCCCP are interesting and challenging combinatorial optimization problems in their own rights, particularly due to their similarities to some travelling salesman- and vehicle routing-family of problems. In this paper, our main focus is to review the existing mathematical programming models and solution methods in the literature, analyse the performance of these models, and identify future research directions. Further, we propose a polynomial-sized and an exponential-sized mixed-integer linear programming model, discuss a number of stronger constraints for cardinality-infeasible-cycle elimination for the latter, and present some preliminary numerical results.

✉ Vicky Mak-Hau
vicky.mak@deakin.edu.au

1 School of Information Technology, Deakin University, Burwood, VIC 3125, Australia

Keywords Kidney exchange · Integer programming · Combinatorial optimization · Healthcare

1 Introduction

According to Kidney Health Australia, approximately 1 in 10 Australians aged 18 or above have indicators of chronic kidney disease. Kidney transplant is one of the treatment options for kidney failure patients. In Australia, patients who are eligible for a kidney transplant often have to wait, on average, for about 4 years for a kidney from a deceased donor (Kidney Health Australia 2015). Some of these patients may have relatives/friends who are willing to donate a kidney to them, but the kidneys may not be compatible due to ABO blood type incompatibility or positive serological cross match. We call a pair of patient and his/her incompatible donor an Incompatible Patient-Donor Pair (PDP). A pool of these PDPs is called a Kidney Exchange Pool.

The idea of kidney exchange provided new hope for kidney failure patients. Suppose we have two pairs of incompatible patient-donors: PDP-A and PDP-B. If the kidney of Donor A is compatible to Patient B, and that of Donor B's is compatible to Patient A, then the two PDPs can exchange kidneys. We call this a *2-way exchange* (or, mathematically, a *2-cycle*). In the early days, kidney exchanges involved 2-way exchanges only. A 3-way exchange (aka a 3-cycle) involves three PDPs, with Donors A, B, and C donating their kidneys to, e.g., Patients B, C, and A respectively. In recent years, kidney exchange becomes more sophisticated with the introduction of multi-way exchanges (see, e.g., Gate (2015) where a 9-way exchange was performed recently), as well as altruistic donors in the kidney exchange pools. With these complications, the underlying mathematical problem becomes more complex and have attracted the community of Integer Programming/Combinatorial Optimization to the research of optimizing kidney exchanges for the best possible outcomes.

Mathematically, the Kidney Exchange Problem (KEP) is a combinatorial optimization problem that can be defined on a directed graph, and typically concerns solutions with either just cycles, or cycles and chains. We call the cycles-only KEP the Cardinality Constrained Multi-cycle Problem (CCMcP) on directed graphs. One can consider the CCMcP as an Asymmetric Travelling Salesman Problem (ATSP) with subtours allowed, yet the subtours are constrained by cardinality, and that it is not necessary to visit all vertices. We call the KEP that allows cycles as well as chains the Cardinality Constrained Cycles and Chains Problem (CCCCP) on directed graphs. In a CCCCP, the chains must start from a pre-determined set of vertices on the graph. Whilst the cycles in a CCCCP are constrained by cardinality in the same way cycles in CCMcP do, the chains are considered to be constrained in some studies but unconstrained in others.

The aim of this paper is to review existing Integer Programming approaches to different variations of the KEP, analyse the performance of various mathematical programming models and solution methodologies, present stronger cuts and preliminary numerical results, and propose future research directions. In Sect. 2, we explain the clinical background for the KEP. In Sect. 3, we list the related combinatorial optimization problems, compare and contrast the most closely related ones. In Sects. 4 and 5, we present existing MILP models, review their respective solution methodologies, present some strong cycle-cardinality violation elimination constraints, and propose a couple of new MILP models for the CCCCP (with preliminary results reported in Sect. 6). In Sect. 7, we review the pre-processing schemes. In Sect. 8, we investigate the multiple criteria nature of the KEP, review current work, and finally in Sect., 9 propose future research directions.

2 Clinical background

The KEP in the form of a CCMcP involves only PDPs and can be explained as follows. Given a pool of PDPs, we would like to optimize the way these kidney exchanges are carried out for the most desired outcome(s). A donor, as soon as the partner patient has received a kidney, can technically exit the program without donating one, as he/she is not legally bound to do so. To avoid this from happening, usually exchanges are carried out simultaneously. As each transplant involves two surgeries, there is a limit as to how many exchanges can be performed at once, due to human resource and logistic reasons. We use K to represent the maximum number of transplants that can be carried out concurrently (i.e., a K-way exchange).

Representing the KEP on a directed graph D, let P be the set of PDPs. If the KEP pool contains only PDPs, then we have $D = (V, A)$, for $V = P$ and $A = \{(i, j) \,|\, i, j \in P,\ i \neq j\}$. Notice that A may not be, and most of the time is simply not, a complete graph. Compatible patient-donor-pairs are in general not considered in the KEP, hence D does not contain loops. Though the inclusion of these pairs has been discussed in recent literature (see, e.g., Gentry et al. 2007), ethicality has been cited as the major issue. A weight w_{ij} is assigned to each arc $(i, j) \in A$. These weights could be used to measure how likely the transplant of a kidney from donor i to patient j is to be successful, or to measure some other aspects of kidney transplants. In some KEPs, the objective is to maximize the total number of kidney exchanges, whereas in other studies, the objective is to maximize the sum of arc weights defined by a solution. In Sect. 8, we will review a number of different objectives discussed or used in existing literature.

In Fig. 1, the PDPs are represented by p_j for $j = 1, \ldots, 14$. A 2-way exchange is formed between p_4 and p_7, and a 4-way exchange is formed amongst p_5, p_6, p_9, and p_{10}. A κ-way exchange (for $\kappa = 2, \ldots, K$) that involves PDPs only is called a *cycle*. Manlove and O'Malley (2012) has demonstrated the benefit of allowing 4-way exchanges, (as opposed to just 2- and 3-way exchanges), in terms of increased number of matches. For example, from a set of clinical data with KEP pool size ranging from 85 to 186, on average, 4 more matching can be achieved by allowing 4-way exchanges. In the U.S., a nine-way kidney exchange was performed successfully in 2015 Gate (2015). This may further improve the outcomes of a KEP. Integer Programming approaches for the CCMcP has been considered in Abraham et al. (2007), Constantino et al. (2013), and Klimentova et al. (2014).

With altruistic donors, the KEP digraph becomes more complex. A sequence of kidney exchanges that begins from an altruistic donor and terminates at the "waitlist" (which usually refers to the deceased donor waiting list) forms a *chain*. In Fig. 1, a

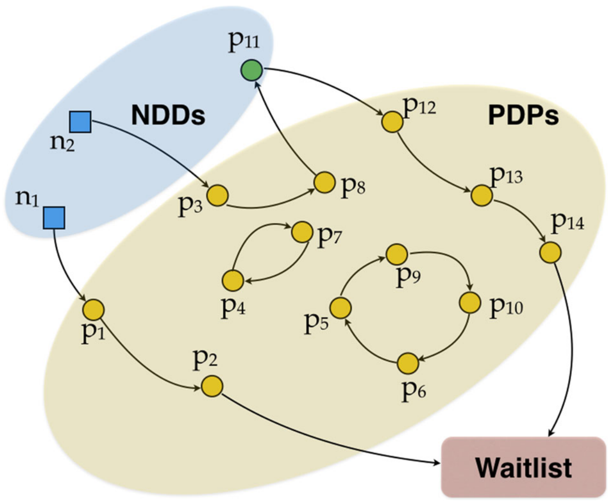

Fig. 1 An example of a kidney exchange pool

chain is formed from altruistic donor n_1 who donates a kidney to patient p_1 whose partner donor in turn donates to patient p_2, and finally the kidney of donor p_2 goes to the deceased donor waiting list. There are two types of chains: *long chains* and *short chains*. In some studies in the literature, a size limit was considered for chains, and we use L to denote it. In other studies, the size of a chain is not constrained. There is a further complication for chains, a long chain can be made up of a sequence of short chains. Take the chain $(n_2, p_3, p_8, p_{11}, p_{12}, p_{13}, p_{14})$ in the same figure as an example: the PDP represented by p_{11} is called a *bridge donor*. What it means is that the transplants from altruistic donor n_2 to patient p_3, from donor p_3 to patient p_8, and from donor p_8 to patient p_{11} are all performed simultaneously. Donor p_{11} will not donate his/her kidney at the same time patient p_{11} receives his/hers, but will wait until the transplants involving PDPs p_{12}, p_{13}, and patient p_{14} are ready. These transplants will then be performed simultaneously. The kidney of donor p_{14} will then go to the deceased donor waiting list. Altruistic donors and bridge donors are collectively called Non-directed Donors (NDDs) in some studies.

In the literature of CCCCP with a size limit for chains (L), some considered L the same as the size limit for cycles, i.e., $L = K$, (see, e.g., Manlove and O'Malley 2012), some considered $L \geq K$, (see, e.g., Glorie et al. 2014), and some considered $L = \infty$, in other words, chain-size unconstrained (see, e.g., Anderson et al. 2015). To describe a CCCCP on a directed graph D, we will have $D = (V, A)$, for $V = N \cup P$, where N is the set of altruistic/bridge donors, and $A = \{(i, j) \mid i \in V,\ j \in P,\ i \neq j\}$. Notice that $\{(i, j) \mid i, j \in N, i \neq j\} = \emptyset$, and that in a CCCCP, a vertex can be in at most one chain or cycle, but not both. A summary table for existing literature is presented in Table 1.

For more detailed descriptions of the history and various forms of the KEP, see, e.g., Gentry et al. (2011). There are of course operations research techniques other than integer programming (e.g., Game Theory or heuristic methods) that have been

Table 1 A summary of existing integer programming exact methods for the KEP

	Max$\|V\|$	Arc Den	K	L	Col Gen	Cut Gen
Abraham et al. (2007)	10,000	Unspecified	3	–	Yes	–
Manlove and O'Malley (2012)	186	19 %	3,4	$=K$	ND	–
Constantino et al. (2013)†						
Low density	500/200/100	20 %	3/4/5,6	NA	–	–
Medium density	400/100	50 %	3/4,5,6	NA	–	–
High density	200/100	70 %	3/4,5,6	NA	–	–
BloodType	1000/200/100	–	3/4/5,6	NA	–	–
Glorie et al. (2014)	500	Unspecified	3	4	Yes	–
	1000	Unspecified	4	6	Yes	–
Klimentova et al. (2014)†	2000	Unspecified	3	NA	Yes	-
	1000	Unspecified	4	NA	Yes	–
	900	Unspecified	5,6	NA	Yes	–
Anderson et al. (2015)						
Formulation 1	1341	19 %	3	∞*	-	Yes
Formulation 2	1341	19 %	3	∞*	ND	Yes
This article	256 /162	4–6,10 %	3/10	∞	–	–

N.B.: (1) In this table, only the implementation reported as numerical results in these articles are listed. Other implementation options discussed in these articles but not presented in numerical results are not listed. (2) A † indicates that problem reduction is performed in pre-processing. (3) For "*", in Anderson et al. (2015), presumably the results presented in the tables therein have $L = \infty$. The authors have also mentioned that they have experimented with $L = 3, 4, 5, 6$ as well. (4) A "-" stands for "Not applicable". (5) "ND" is short for "Not Described". (6) "Max$|V|$" is the largest size of $|V|$ solved to optimality in the respective numerical experiments. (7) "Arc Den", "Col Gen", and "Cut Gen" are short forms for "Arc Density", "Column Generation" and "Cut Generation" respectively. For arc density, the number describes the highest arc density in the test instances. (8) For a row with multiple values in Max$|V|$ and K, e.g., Max$|V| = 500/200/100$ and $K = 3/4/5, 6$, we mean that for $K = 3$, $K = 4$, and $K = 5$ or 6, the largest test instances solved to optimal has $|V|$ =500, 200, and 100 respectively studied in the literature for various aspects/forms of the KEP. We refer readers to, e.g., Biró et al. (2009), Zenios et al. (2000), Chen et al. (2012), Dickerson et al. (2012), Ashlagi et al. (2011), and Gentry et al. (2009).

The CCMcP and the CCCCP are interesting combinatorial optimization problems in their own rights. Aside from the clinical application, the CCMcP is also studied in the context of barter exchange (see, e.g., Abraham et al. 2007 and Roth et al. 2007). The aim of this paper, however, is to focus on exact methods for the KEPs, namely integer programming formulations, solution methodologies, and their numerical performances in the context of constrained cycles and chains problems on directed graphs, and propose future research directions.

3 Closely related combinatorial optimization problems

The CCCCP concerns cardinality-constrained multi-cycle and multi-chain simultaneously, to the best of our knowledge, there is no previous studies in such type of problems in the literature of combinatorial optimization, except for Glorie et al. (2014) and Anderson et al. (2015). On the other hand, for the CCMcP, there are a number of closely related constrained cycle(s) problems. There are, however, some major differences, namely:

- Some constrained cycle problems studied in the literature concern only a single cycle, see, e.g., the Cardinality Constrained Travelling Salesman Problem (Cao and Glover 1997); and the Cardinality Constrained Knapsack Problem defined on undirected graphs, (Bauer et al. 2002). The CCMcP and CCCCP allow multiple cycles (as well as multiple chains in the latter).
- Whenever multiple cycles are involved in those previously studied constrained cycle problems, usually it is required that all vertices be visited, see, e.g., the Cardinality Constrained Covering Travelling Salesman Problem, (Patterson and Rolland 2003), whereas neither the CCMcP nor the CCCCP has this restriction.
- Some constrained single- or multi-cycle problems require that for the cycles, exactly K vertices must be used, see, e.g., the K-cycle Problem (Nguyen and Maurras 2001 and Hartmann and Özlük 2001). In the case of CCMcP or the CCCCP, a feasible solution can contain cycles with less than K vertices.

Perhaps the closest are the Vehicle Routing Problems with Capacity Constraints (see, e.g., Fischetti et al. 1998; Toth and Vigo 2002; Baldacci et al. 2010; Cornuejols and Harche 1993), and the Asymmetric Travelling Salesman Problem with Replenishment Arcs (RATSP), (see, e.g., Mak and Boland 2000, 2006, 2007). In particular, when all customers in the former have unit demand, or all vertices in the latter have unit weight, these two classic combinatorial optimization problems closely resemble the CCMcP. (Readers who are unfamiliar with the terminology used in these classic combinatorial optimization problems can find relevant definitions in the papers cited above).

One important aspect is that subtours are not allowed in Vehicle Routing- or Travelling Salesman-family of problems. Therefore, even though several classes of strong cardinality violation elimination constraints have been developed in the literature for these problems, those constraints in general eliminate subtours as well, hence are not valid for eliminating cardinality-violated cycles for the CCMcP or the CCCCP, as we *do* allow cycles with smaller sizes, as long as they are no greater than K. For a similar reason, the original Subtour Elimination Constraint (SEC) of Miller-Tucker-Zemlin (MTZ) (Miller et al. 1960) cannot be applied for the CCMcP and the CCCCP in eliminating cardinality violation in cycles either. In the Vehicle Routing Problems with Capacity Constraints, as all cycles start and finish at the depot, it is possible to derive a set of polynomial-sized constraints with a time-stamp nature, in the manner of MTZ, for eliminating cardinality violations by assigning a time-variable for all vertices except for the depot. Similarly for the RATSP, where a weight feasible ordinary path begins/ends at the end/start of a replenishment path respectively. For the cycles in both a CCMcP or a CCCCP, on the other hand, since they do not have a "start" or an "end", a MTZ-type cardinality violation constraint cannot be applied. For the chains in a CCCCP, however, we are able to produce a set of MTZ-type constraints for cycle-elimination and cardinality violation-elimination within chains, as we shall explain in detail in Sect. 4.

As mentioned earlier, one can view the CCMcP as an ATSP with subtours allowed, but constrained in cardinality and that not all vertices must be visited. We also looked at cardinality constrained Assignment Problems, but have found nothing closely related to the CCMcP. The K-assignment Problem (see, e.g., Dell'Amico and Martello 1997 and Bai 2009) requires that the total assignment is equal to K. For a more thorough exposition of cardinality constrained cycle and path problems, see, e.g., Kaibel and Stephan (2007), Kaibel and Stephan (2010).

4 Polynomial-sized formulations

In Constantino et al. (2013), two polynomial-sized formulations were presented for the CCMcP: the Edge Assignment (EA) model and the Extended Edge (EE) model. The EA model uses a set of binary variables to indicate whether an arc is used in the solution and another set of binary variables to indicate whether a vertex belongs to a particular cycle. These two sets of variables are then linked together by logic constraints to indicate the arcs involved in each of the cycles. The Extended Edge (EE) model, on the other hand, makes $|V|$ copies of the directed graph D, and a three-index binary variable is used to determine whether an arc is used in a particular cycle. Pre-processing for problem reduction and symmetry elimination were proposed for both models. More discussions on pre-processing will be provided in Sect. 7. In this section, we will focus our discussions on the EE model as it was reported to be computationally more superior. Since the EE model concerns only cycles and no chains, we have $V = P$, (recall that P is the set of all PDPs).

Let $D = (V, A)$ be cloned into $|V|$ copies, and let $\mathcal{L} = \{1, \ldots, |V|\}$. We require that each copy of D contains at most one cycle. Let x_{ij}^{ℓ} be a binary variable with $x_{ij}^{\ell} = 1$ indicating arc $(i, j) \in A$ is used in cycle $\ell \in \mathcal{L}$. Recall that the weight of an arc $(i, j) \in A$ is denoted by w_{ij}. The full ILP formulation for the EE model is given as follows.

Model 1 *Polynomial-sized formulation–Constantino et al. (2013)*

$$\max \sum_{\ell\in\mathcal{L}} \sum_{(i,j)\in A} w_{ij} x_{ij}^{\ell} \tag{1}$$

$$s.t. \quad \sum_{j:\,(j,i)\in A} x_{ji}^{\ell} = \sum_{j:\,(i,j)\in A} x_{ij}^{\ell}, \qquad \forall i \in V,\ \forall \ell \in \mathcal{L} \tag{2}$$

$$\sum_{\ell\in\mathcal{L}} \sum_{j:\,(i,j)\in A} x_{ij}^{\ell} \le 1, \qquad \forall i \in V \tag{3}$$

$$\sum_{(i,j)\in A} x_{ij}^{\ell} \le K \qquad \forall \ell \in \mathcal{L} \tag{4}$$

$$\sum_{j:\,(i,j)\in A} x_{ij}^{\ell} \le \sum_{j:\,(i,j)\in A} x_{\ell j}^{\ell} \qquad \forall i > \ell,\ \forall \ell \in \mathcal{L} \tag{5}$$

$$\sum_{j:\,(i,j)\in A} x_{ij}^{\ell} = 0 \qquad \forall i < \ell,\ \forall \ell \in \mathcal{L} \tag{6}$$

$$x_{ij}^{\ell} \in \{0,1\}, \qquad \forall (i,j) \in A, \forall \ell \in \mathcal{L} \tag{7}$$

Constraint (2) ensures the flow balance of a vertex–if a patient of a PDP receives a kidney, then the donor will donate his/her kidney to another patient in the pool. Constraint (3) guarantees that no more than one kidney transplant is involved for each PDP. Constraint (4) makes sure that the cardinality of each cycle is not more than K. Constraints (5) and (6) are able to eliminate a great deal of symmetry in the IP model induced by permutation of cycle indices. Symmetry elimination is achieved by restricting that the index of a cycle to be exactly the smallest vertex-index among all vertices involved in the cycle.

Constantino et al. (2013) compared the EA model, the EE model, together with two exponential-sized formulations of Roth et al. (2007) on four classes of randomly generated test instances. The first three classes of test instances are randomly generated directed graphs with arc densities 20, 50, and 70 %. The fourth class of test instances is the *blood-type* instances generated according to Saidman et al. (2006). All tests were carried out using CPLEX 12.2. As we shall discuss later, both formulations of Roth et al. (2007) are exponential in size–the arc-based model has exponentially many constraints and the cycle-based model has exponentially many variables. Since neither column generation nor cut generation were discussed in the paper, and only $K = 3, \dots, 6$ were tested, presumably the constraints and columns are exhaustively generated and added to the IP solver. For large-scale problem instances, only the cycle-based formulation and the EE model were compared. For blood-type test instances, and those with 20 % arc density, the cycle-based model performed better as cycles are relatively scarce, and solved more problems to optimality, particularly for $K = 3, 4,$ and 5. For $K = 3$, the cycle-based model solved problems with up to $|V| = 1000$ for the blood-type test instances and up to $|V| = 500$ for low density test instances. For $K = 4$, the cycle-based model solved to optimality the same two classes of test instances with up to $|V| = 200$.

The EE model, on the other hand, performed better for problems with medium (50 %) and high (70 %) arc densities, particularly for $K = 5$ and 6. In general, the method solved to optimality problems of size $|V| = 100$ in 70–80 % of these two classes of test instances, within a computation time limit of 1800 seconds. Further, for the EE model, Constantino et al. (2013) applied problem reduction pre-processing that is capable of removing a large portion of variables. The same pre-processing algorithm cannot be applied for the two formulations of Roth et al. (2007). Constantino et al. (2013) also briefly described how their polynomial-sized ILP model can be modified to accommodate altruistic donors, namely, by considering chains as cycles, and setting different cardinality constraints for cycles that contain an altruistic donor and those that do not. The vertices representing altruistic donors are also required to be cloned into multiple copies. The model, however, was not numerically tested.

4.1 A natural extension–polynomial size formulation for the CCCCP

We now propose an alternative, more compact extension of Constantino et al. (2013) to allow chains. Essentially we split the binary arc variable x_{ij}^{ℓ} used in Constantino et al. (2013) into two: $y_{ij} \in \{0, 1\}$ and $u_{ij}^{\ell} \in \{0, 1\}$. By doing so, we are able to determine whether an arc belongs to a chain or a cycle. We use N to denote the set of NDDs. Notice that $N \cap P = \emptyset$. Let:

- τ be an auxiliary terminal node (that serves as the deceased donor "waitlist");
- $A' = A \cup \{(i, j) \mid i \in V, j \in P \cup \{\tau\}\}$, with $A = \{(i, j) \mid i, j \in P\}$;
- D be redefined as $D = (V \cup \{\tau\}, A')$;
- $y_{ij} = 1$ indicating the arc $(i, j) \in A'$ forms part of a chain, $y_{ij} = 0$ otherwise; and
- $u_{ij}^{\ell} = 1$ indicating the arc $(i, j) \in A$ forms part of the ℓth cycle, $u_{ij}^{\ell} = 0$ otherwise;
- t_i, for $i \in V$, a continuous variable as "time stamp" for vertex j should it be part of a chain.

(Notice that as there is no need to index the chains for cycle- or cardinality violation-elimination, our model will not suffer from symmetry by index permutation). Now, we modify the objective function and replace Constraint (2) by Constraints (9) and (10), and replace Constraint (3) by Constraints (11) and (12).

Model 2 *The polynomial-sized SPLIT formulation*

$$\max z = \sum_{(i,j)\in A'} w_{ij} y_{ij} + \sum_{\ell\in\mathcal{L}} \sum_{(i,j)\in A} w_{ij} u_{ij}^{\ell} \tag{8}$$

$$\sum_{j\in P:\,(i,j)\in A} u_{ij}^{\ell} = \sum_{j\in P:\,(j,i)\in A} u_{ji}^{\ell}, \qquad \forall i \in P,\ \forall \ell \in \mathcal{L} \tag{9}$$

$$\sum_{j\in P\cup\{\tau\}:\,(i,j)\in A'} y_{ij} = \sum_{j\in V:\,(j,i)\in A'} y_{ji}, \qquad \forall i \in P \tag{10}$$

$$\sum_{j\in P\cup\{\tau\}:\,(i,j)\in A'} y_{ij} \leq 1, \qquad \forall i \in N \tag{11}$$

$$\sum_{j\in P\cup\{\tau\}:\,(i,j)\in A'} y_{ij} + \sum_{\ell\in\mathcal{L}} \sum_{j\in P:\,(i,j)\in A} u_{ij}^{\ell} \leq 1, \qquad \forall i \in P \tag{12}$$

$$\begin{array}{rcll} \multicolumn{3}{r}{\textit{Constraints } (4)-(7),} & \textit{with } x_{ij}^{\ell} \textit{ replaced by } u_{ij}^{\ell} \\ t_i - t_j + |P|y_{ji} + (|P|+2)y_{ij} & \leq & |P|+1, & \forall i \in V,\ j \in P \cup \{\tau\} \quad (13) \\ t_i & = & 0, & \forall i \in N \quad (14) \\ t_i & \geq & 0, & \forall i \in P \cup \{\tau\} \quad (15) \\ t_\tau & \leq & |P|+1. & (16) \end{array}$$

Notice that without (13)–(16), the chain variables y_a, for all $a \in A'$, may themselves induce cycles. Subtour Elimination Constraints (SECs), in particular those strong SECs in the context of Asymmetric Travelling Salesman Problem (ATSP) can certainly be implemented. There will, however, be exponentially many SECs, hence cutting plane is expected within a branch-and-cut framework for exact solutions. For a formulation that is polynomial in size, on the other hand, we can use MTZ-type constraints. Let t_i be the time-stamp of vertex i, for $i \in V$, and we require that should vertex j be visited immediately after vertex i in a chain, we must have $t_j \geq t_i + 1$. This can be achieved by Constraint (13). Constraints (14)–(16) are bound constraints.

The benefit of using the MTZ-type formulation for subtour elimination in chains is that it can be easily modified to accommodate cardinality restrictions in chains. E.g., if the maximum chain size, not including τ, is L, we simply replace (16) by:

$$t_\tau \leq L + 1. \tag{17}$$

(If chain size includes τ, then the right-hand-side of (17) should be just L.) In Anderson et al. (2015), a Constantino et al. (2013)-style graph cloning idea is used to model the cardinality limit on L, however no numerical results were presented in this respect. In Tables 3 and 5, we have compared such constraints proposed in Anderson et al. (2015) (in columns under "AND-L") and the MTZ-style constraints we proposed above (in columns under "SPLIT-MTZ"), both incorporated within Model 6. We can see that the MTZ-style size constraints on L appear to be computationally more efficient.

5 Exponential-sized formulations

In the context of CCMcP, most MIP models are either arc based, (using a binary variable for each arc in A), or cycle/chain based, (using a binary variable for each feasible cycle/chain). There are exponentially many cardinality-infeasible-cycle elimination constraints with the former but exponentially many variables with the latter. The firsts of both arc- and cycle-based formulations are proposed in Roth et al. (2007).

5.1 Cycle/chain-based formulation

We first look at the cycle/chain-based formulation. Let:

- Γ be the index set of all cycles with size no more than K (and all chains with size no more than L);
- z_γ be a binary variable with $z_\gamma = 1$ if cycle/chain $\gamma \in \Gamma$ is selected, and 0 o.w.;
- $V_\gamma \subseteq V$ be the set of vertices in cycle/chain γ;

– $\rho_\gamma = \sum_{e \in E_\gamma} w_e$, for $E_\gamma \subseteq E$ be the total weight of edges involved in cycle/chain γ.

The cycle/chain-based MIP formulations is given as follows (see, e.g., Roth et al. (2007), Abraham et al. (2007) in the context of CCCmP; and Glorie et al. (2014) and Anderson et al. (2015) in the context of CCCCP).

Model 3 *Cycle/chain-based formulation–Roth et al. (2007)*

$$\max \sum_{\gamma \in \Gamma} \rho_\gamma z_\gamma \tag{18}$$

$$s.t. \sum_{\gamma \,:\, i \in V_\gamma} z_\gamma \le 1, \qquad \forall i \in V. \tag{19}$$

Abraham et al. (2007) were amongst the firsts to implement the cycle-based exponential size algorithm for the CCMcP, and solved test instances with $K = 3$. A branch-and-price (BNP) method is implemented, though the pricing problem is itself NP complete and is solved by a depth-first tree-search, essentially a complete enumeration on the directed graph D to find the cycle with the most positive price. A single column is added at each iteration of column generation, (thought it was also mentioned that feasible cycles are identified along the way too). Column initialization is performed by taking the union of cycles generated by the following three methods. (1) Randomly select an uncovered vertex, in a greedy manner, form a cycle to cover it as well as other uncovered vertices. (2) Solve the maximum-weight matching problem (in $\mathcal{O}(|V|^3)$ time) to obtain a set of 2-cycles. (3) Perform a random walk on the directed graph, to produce a random collection of feasible cycles. The size of the LP is controlled in the way that a predetermined threshold is used to control the number of columns in the LP. When new columns are added, some columns may be removed if the total number of columns has exceeded this threshold. The columns that are used for branching or those that are currently having a non-zero value, however, will not be removed. To speed up the branch-and-bound method, an upper bound is obtained from the polynomially-solvable cardinality-unrestricted directed matching problem, (essentially the Assignment Problem (AP) relaxation). It was mentioned that for test instances of very large scale, however, as the number of arcs is very large, column generation is required even for the AP relaxation. Occasionally, the restricted LP was solved by adding back the integrality constraints and calling CPLEX primal heuristics to obtain a primal solution. A rounding-type of primal heuristic was also implemented. The former was reported to be producing better lower bounds than the latter. Abraham et al. (2007) reported experimental results with test instances with up to $|V| = 10,000$ (with arc density unspecified), for $K = 3$ only, with an average run time of roughly between 3,000 and 3,500 seconds. (It is hard to be precise as the results are presented in the form of a graph).

Manlove and O'Malley (2012) used a similar model for the CCCCP that include "short chains" with the same size limit as cycles. Incoming arcs to the NDDs are allowed and in the context of KEP, these are considered as kidneys going to the deceased donor waiting list, hence a chain is considered the same as a cycle. As only

$K = 3$ is extensively tested, presumably all columns (cycles) are included in the ILP. From their clinical data set, we have observed that the proportion of variables in terms of number of feasible 2- or 3-cycles versus the number of vertices and arc density varies substantially. E.g., for a test instance with 147 vertices and 901 arcs, there are only four 2-cycles and four 3-cycles, but for another problem instance with 141 vertices and 1248 arcs, there are as many as fifty-five 2-cycles and a hundred sixty-six 3-cycles. Amongst the test instances, the largest $|V|$ is 186, and the largest $|A|$ is 1263. No computation times are reported.

Glorie et al. (2014) extended the column generation idea of Abraham et al. (2007) for the CCCCP by including variables (columns) for chains as well, and presented an "exact algorithm" for solving the pricing problem with polynomial complexity $\mathcal{O}(|L||V||A|)$, under the condition that the reduced cost of a cycle can be expressed as a linear function of arc weights. A new graph (the *reduced cost graph*) will be formed using these *reduced arc weights*. These arc weights can be negative or non-negative, and may even contain a negative cycle. The pricing algorithm described is that for each pair of vertices, one will be the source, and the other the sink. A shortest path-like algorithm is executed, but with only K and L steps carried out for cycles and chains respectively. It was unclear as to how a shortest path from the source to the sink can be guaranteed that uses no more than K or L vertices. In any case, it appears that for cycle generation, the shortest path-like algorithm will return a sequence of no more than K vertices, and the total arc weights is added to the weight of a back arc from the last vertex to the first vertex. If the sum is less than 0, then a negative reduced cost cycle (i.e., a new column) is produced. (Although, it appears that the way the back arc is added after the shortest path is found, the new column generated is not necessarily the one with the most negative reduced cost). A similar strategy is implemented for new columns that represent chains. As for the possible existence of negative cost cycles in the reduced cost graph, presumably those with less than K or L vertices will themselves be new columns, though again not necessarily with the most negative reduced cost.

Two branching schemes were proposed and implemented. One of which is the common practice of selecting the variable that is closest to 0.5, creating two branches, with one setting the variable to 0 and the other, 1. The other branching scheme was to select the vertex with the largest number of outgoing arcs, list all arc values in non-descending order, put arcs into set S_1, until the sum reaches 0.5. The rest of these outgoing arcs will be included in a second set S_2. One branch is created for each of S_1 and S_2 wherein all variables in the set are set to 0. It was reported that both branching schemes performed rather similarly in the numerical experiments. The test set used in Glorie et al. (2014) are simulated data using the simulator described in Saidman et al. (2006), it contains $|V|$ ranging from 10 to 500, with two specific cardinality settings for the cycles and chains: (i) $K = L = 3$; and (ii) $K = 4$, $L = 6$. For the former, the fastest computation time for the largest problem instances, $|V| = 500$, is less than 22 seconds, and for the latter, less than 96 seconds.

Klimentova et al. (2014), on the other hand, proposed a decomposition by ℓ-model for the CCMcP that combines column generation with the graph-cloning idea of Constantino et al. (2013), where a column generation subproblem is solved for each copy of D (denoted by $D^\ell = (\tilde{V}^\ell, \tilde{A}^\ell)$, for $\ell = 1, \ldots, |V|$). With the column generation subproblem for D^ℓ, the only cycles that are required to be generated are those that involve ℓ and some other vertices with indices greater than ℓ. The column generation subproblem is presented as an IP model, but was in fact solved using a Greedy Heuristic. The heuristic starts from vertex ℓ, inserts the vertex on D^ℓ with an arc that is most "profitable". If a back arc to ℓ exists, then a cycle is formed, and it will be returned as a new column. Otherwise, the next vertex with the most profitable arc will be added to the path obtained so far. If a back arc exists, a cycle is formed; otherwise, if an outgoing arc exists for this vertex, the procedure will be continued, and if not, then backtracking to the previous vertex has to be performed. The heuristic is not polynomial in complexity as backtracking is required, so the computation time when K is large can be substantially longer. Notice that the heuristic method favours columns that represent cycles of smaller cardinality.

Now, the solution for each subproblem ℓ will provide a "column" for the Restricted Master IP, should it exists, hence providing a number of new columns at a time. Some variations of the method were implemented, e.g., after column generation ceases in the root node, the resulting restricted LP is solved as an IP rather than an LP relaxation. It was briefly mentioned that meta-heuristic approaches (namely Local Search and Variable Neighbourhood Search) were also implemented for producing lower bounds in the root node or in all nodes of the BNB tree. Implementation details for those search heuristics were not specified. A weighted objective function has been tested too, with a much higher weight given to the number of matches, and a lighter weight given to the number of cycles in a solution. The motivation for the latter is that as the number of cycles is maximized, it is likely that the size of each cycle is smaller (though this cannot be guaranteed). Some impressive numerical results were presented. Test instances with up to 2000 nodes were solved for $K = 3$ (with average run time for the fastest algorithm 51.4 seconds), up to 1000 nodes for $K = 4$ (with average run time of the fastest algorithm 82.4 seconds), and up to 900 nodes for $K = 5$ and $K = 6$ (with average run times for the best algorithms 125.4 and 214.6 seconds respectively). It would be nice if there is numerical evidence of the improvement brought by the decomposition by ℓ when compared to a standard column generation without it wherein, e.g., a similar Greedy Insertion heuristic or some pricing method similar to the one described in Glorie et al. (2014) is implemented.

In the context of CCCCP, Anderson et al. (2015) proposed a model that contains an exponential number of constraints (for subtour elimination of the arc variables used in chains) and an exponential number of variables (one for each feasible cycle). Column generation was not described in Anderson et al. (2015). As the test problem instances are for $K = 3$ only, presumably all feasible 2- and 3-cycles are exhaustively generated. More details on this formulation will be discussed in Model 7.

5.2 Arc-based formulations

The arc-based formulation has more varieties, mainly due to the differences in problem structures between CCMcP and CCCCP, and the differences in restrictions for cycles and chains. We begin with formulations for the CCMcP. We define $\pi = (i_1, \ldots, i_{K+1})$ to be a *minimal infeasible path* or a *minimal cardinality violation path*. Let

– Π be the index set of all minimal infeasible paths;
– x_{ij}, for all $(i, j) \in A$, be a binary decision variable with $x_{ij} = 1$ if arc (i, j) is used, and 0 o.w.; and
– w_{ij} be the weight of arc $(i, j) \in A$.

Model 4 *Arc-based formulation—Roth et al. (2007)*

$$\max \sum_{(i,j)\in A} w_{ij}x_{ij} \tag{20}$$

$$\text{s.t.} \sum_{j:\,(i,j)\in A} x_{ij} = \sum_{j:\,(j,i)\in A} x_{ji}, \qquad \forall i \in V \tag{21}$$

$$\sum_{j:\,(i,j)\in A} x_{ij} \le 1, \qquad \forall i \in V \tag{22}$$

$$\sum_{(i,j)\in \pi} x_{ij} \le K - 1, \qquad \forall \pi \in \Pi \tag{23}$$

$$x_{ij} \in \{0, 1\}, \qquad \forall\, (i, j) \in A. \tag{24}$$

Without Constraint (23), the problem is an Assignment Problem (AP), and will give us subtours that may or may not be of sizes greater than K. As AP can be solved in polynomially time, as a future research direction, one may attempt a branch-and-bound method with each node solving an AP-based Lagrangean Relaxation. The reason is, in Mak and Boland (2007), a similar exact algorithm was implemented for the RATSP. From the computational results, it appeared that the method of Mak and Boland (2007) is computationally much more efficient than the branch-and-price-and-cut algorithm of Boland et al. (2000) for the RATSP.

For CCCCP, however, Constraint (23) is invalid for $L > K$ if the same binary variable x_{ij} is used for indicating an arc (i, j) is used in either a chain or a cycle, as is the arc-based model presented in Anderson et al. (2015), wherein $L = \infty$ is considered. (Though $L = 3, 4, 5, 6$ were mentioned in the description of the numerical experiments, judging from the text and the numerical results reported in the two tables, it is likely that $L = \infty$ was implemented.) Let $C = (i_1, \ldots, i_{|C|})$, for $|C| > K$, be a cycle with cardinality constraint violated, and Λ be the set of all cardinality-violated cycles. Let $A^* = A \cup \{(i, j) \mid i \in N, j \in P\}$, where $A = \{(i, j) \mid i, j \in P, i \neq j\}$.

Model 5 *Arc-based formulation–Anderson et al. (2015)*

$$\max \quad (20)$$

$$\text{s.t.} \sum_{j\in P:\,(i,j)\in A^*} x_{ij} \le 1, \quad \forall i \in N \tag{25}$$

$$\sum_{j\in V:\,(i,j)\in A^*} x_{ij} \le \sum_{j\in V:\,(j,i)\in A^*} x_{ji} \le 1, \quad \forall i \in P \tag{26}$$

$$\sum_{j=1}^{|C|-1} x_{i_j,i_{j+1}} + x_{i_{|C|},i_1} \le |C| - 1, \quad \forall C \in \Lambda$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in A^*. \tag{27}$$

We can see that (25) and (26) are modified from (22) and (21), and that Constraint (27) is equivalent to the weak SEC for the ATSP. Due to the fact that the same variable is used irrespective of an arc being part of a chain or a cycle, only the weak cardinality-infeasible-cycle elimination constraints (i.e., (27)) can be applied. In fact, the smaller K is, the more such constraints there will be, and thus cut-generation will be required. Stronger constraints, e.g., (23) are invalid, as we can have any sequence of $K + 1$ or more distinct vertices used in a feasible solution, as long as they belong to a chain. To solve the IP, Anderson et al. (2015) described that initially constraints in (27) are relaxed, violated constraints are then identified from the integer solution and added to the IP model. The IP is then re-optimized with these additional constraints. This procedure is repeated until an IP solution with no cardinality violated cycles is obtained.

Adopting the idea of the split model we proposed in Model 2, for strong cardinality-infeasible-cycle elimination, instead of using u^{ℓ}_{ij} to represent an arc (i, j) being used in Cycle ℓ, we simply have a binary variable u_{ij}, for each $(i, j) \in A$, with $u_{ij} = 1$ indicating arc (i, j) used in a cycle. The full model is as follows.

Model 6 *The exponential-sized SPLIT formulation*

$$\max \sum_{(i,j)\in A'} w_{ij} y_{ij} + \sum_{(i,j)\in A} w_{ij} u_{ij}$$

$$\text{s.t. } \textit{Constraints } (10) - (11),\ (13) - (16) \tag{28}$$

$$\textit{Constraints } (23)\ (\textit{with } x_{ij} \textit{ replaced by } u_{ij}) \tag{29}$$

$$\sum_{j\in P\cup\{\tau\}:\ (i,j)\in A'} y_{ij} + \sum_{j\in P:\ (i,j)\in A} u_{ij} \le 1, \quad \forall i \in P \tag{30}$$

$$\sum_{j\in P:\ (i,j)\in A} u_{ij} = \sum_{j\in P:\ (j,i)\in A} u_{ji}, \quad \forall i \in P. \tag{31}$$

The benefit of using a SPLIT model is that now stronger version of cardinality-infeasible-cycle elimination constraints can be used, e.g., we can add (23) for each minimal cardinality violation path, $\pi \in \Pi$. When K and $|V|$ are small, one can exhaustively enumerate all constraints into the IP model–there are $\mathcal{O}(|V|^{K+1})$ of these constraints, (compared to the number of constraints in (27) which is exponential even when K is small). Notice that (23) can be further strengthened. Let:

- $V(\pi) = \{i_1, \ldots, i_{K+1}\}$ be the set of vertices in a minimal violation path π;
- $u(\mathcal{F}) = \sum_{s=1}^{K} \sum_{t=s+1}^{K+1} u_{i_s,i_t}$; and
- $u(\mathcal{B}) = \sum_{s=3}^{K} \sum_{t=2}^{s-1} u_{i_s,i_t}$.

Lemma 5.1 *The constraints:*

$$u(\mathcal{F}) \leq K-1, \tag{32}$$

$$u(\mathcal{F}) - \sum_{s=1}^{K-1} u_{i_s,i_{K+1}} + \sum_{j' \notin V(\pi)} u_{i_K,j'} \leq K-1, \tag{33}$$

$$\sum_{(i,j)\in\pi} u_{ij} + u(\mathcal{B}) + \sum_{j' \notin V(\pi)} u_{i_K,j'} \leq K-1, \tag{34}$$

for all minimal cardinality infeasible path $\pi \in \Pi$, are valid.

We present the proofs of validity and other polyhedral results in a companion paper Mak-Hau (2015). From the Linear Programming relaxation (LPR) upper bounds presented in Table 2, we can see that whilst the lifted constraints hardly improved the LPR upper bounds for the objective of maximizing number of matches, Constraints (33) and (34) did produce an impressive improvement to the LPR for the objective of maximizing sum of arc weights. Constraint (32), on the other hand, did not improve (23) significantly. With the lifted constraints, the LPR with either (33) or (34) is stronger than the LPR of the polynomial-sized Model 2 that is based on the Constantino et al. (2013) model together with a MTZ -type chain cardinality constraint.

Anderson et al. (2015) also presented a second model for the CCCCP, a mixed arc-and-cycle-based formulation that contains exponentially many variables as well as exponentially many constraints. Let Ω_K be the set of all cycles with cardinality no more than K. Recall that $A^* = A \cup \{(i,j) \mid i \in N, j \in P\}$, where $A = \{(i,j) \mid i,j \in P, i \neq j\}$. The mixed formulation uses a binary variable y_a, one for each $a \in A^*$ with $y_a = 1$ indicating arc a is used as part of a chain; and z_γ a binary variable, one for each cardinality feasible cycle $\gamma \in \Omega_K$ with $z_\gamma = 1$ indicating cycle γ is used in the solution. (Notice that in this model there is no need for a terminal node τ). Let $\delta^-(S) = \{(j,i) \in A^* \mid j \in \bar{S}, i \in S\}$; and $w_\gamma = \sum_{a \in A(\gamma)} w_a$, where $A(\gamma)$ are the arcs used in γ, for any $\gamma \in \Omega_K$.

Model 7 *Mixed formulation—Anderson et al. (2015)*

$$\max \sum_{(i,j)\in A^*} w_{ij} y_{ij} + \sum_{\gamma\in\Omega_K} w_\gamma z_\gamma \tag{35}$$

$$\text{s.t.} \sum_{j\in P:\,(i,j)\in A^*} y_{ij} \leq 1, \qquad \forall i \in N \tag{36}$$

$$\sum_{j\in P:\,(i,j)\in A} y_{ij} + \sum_{\gamma\in\Omega_K(i)} z_\gamma \leq \sum_{j\in V:\,(j,i)\in A^*} y_{ji} + \sum_{\gamma\in\Omega_K(i)} z_\gamma \leq 1, \qquad \forall i \in P \tag{37}$$

$$\sum_{(j,i)\in\delta^-(S)} y_{ji} \geq \sum_{j\in V:\,(j,i)\in A^*} y_{ji}, \qquad \forall S \subseteq P,\ i \in S \tag{38}$$

The model contains exponentially many constraints in (38), and that the number of z_γ variables is of order $\binom{|P|}{K}$. Constraint (38) is a cut-set type SEC borrowed from the ATSP. Should one considers larger K, column generation would be necessary, but at the same time, due to the chain variables, there is also an exponential number of subtour elimination constraints. Hence with larger K, the model would require both

Table 2 A comparisons of the linear programming relaxation (LPR) upper bound with various cardinality-infeasible-cycle elimination constraints

Test instances	(23)	(32)	(33)	(34)	(33) + (34)	EE-MTZ	Opt
Linear programming relaxation upper bounds—max number of matches							
CLIN53	53	53	53	53	53	53	51
RAN106	105	105	105	105	105	105	–
CLIN166	165	165	164.333	164.583	164.333	164.833	–
RAN255	253	253	252.55	252.55	252.55	OOM	–
Linear programming relaxation upper bounds—max weighted sum							
CLIN53	196.536	196.325	194	193.857	193.5	198.454	166
RAN106	823.436	823.184	812.938	813.04	812.934	814.106	–
CLIN166	1230.274	1230.145	1208.595	1208.558	1208.123	1224.226	–
CLIN166	1806.791	1806.494	1770.886	1771.076	1770.68	OOM	–

column and cut generation. Depending on how implementation is carried out, if the structure of one destroys that of the other, the separation and pricing problems may not be polynomially solvable.

In their implementation, only $K = 3$ is reported, and column generation was not discussed. Presumably all columns were exhaustively generated and included in the IP model. In their numerical experiments, Anderson et al. (2015) presented results for two sets of test instances. The first set of test instances has the number of PDPs ($|P|$) ranging from 152 to 389, the number of NDDs ($|N|$) ranging from 1 to 11, and $|A|$ ranging from 1,109 to 13,711. In all cases, Model 7 appears to excel Model 5, with the former solving all test instances to optimality within 5 seconds, whilst the latter was unable to solve some test instances to optimality even after 1200 second of run time. However, in the second test set, which contains two test instances, one of which contains 47 NDDs, 931 PDPs, with $|A| = 190,820$; and the other 162 NDDs, 1,179 PDPs, with $|A| = 346,608$, Model 5 solved the problem within 3 seconds, but Model 7 took 104 seconds to solve the first instance, and 314 to solve the second.

To deal with constrained chain size, Anderson et al. (2015) proposed to use an “edge extension” approach similar to that of Constantino et al. (2013) in the manner that multiple copies of the graph is cloned, and that each copy of the graph will contain at most one chain. In specific, one can add the following constraints to Model 7.

$$\sum_{\ell \in N} y_{ij}^{\ell} = y_{ij}, \qquad \forall (i,j) \in A^* \tag{39}$$

$$\sum_{(i,j)\in A^*} y_{ij}^{\ell} \le L, \qquad \forall \ell \in N \tag{40}$$

$$\sum_{(i,j)\in \delta^+(i)} y_{ij}^{\ell} \le \sum_{(j,i)\in \delta^-(i)} y_{ji}^{\ell} \le 1, \qquad \forall \ell \in N,\ i \in P. \tag{41}$$

No results are presented for constrained L in Anderson et al. (2015). We have numerical evidence, (see Tables 3 and 5), that the MTZ-style size constraints on L that we proposed in this paper appear to be computationally more efficient.

6 Preliminary numerical testing

We experimented with three models on test instances with different parameters. In Tables 3 and 4, we have tested the SPLIT model for the CCCCP, (Model 6); a polynomial-sized MILP formulation for the CCCCP, (Model 2), and another formulation obtained from Model 6 but with chain restriction modelled in the manner described in Anderson et al. (2015), i.e., (39)–(41). These three models are referred to as the SPLIT-MTZ, EE-MTZ, and AND-L respectively in the tables. We did not implement any problem reduction schemes. All models are coded and solved using IBM ILOG CPLEX 12.5. No cut or column generation were implemented.

As we can see from the results in Tables 3 and 4, the polynomial size formulation for the CCCCP (i.e., EE-MTZ) is a clear winner in most test instances, in particular for modest to moderate sized instances. As the size of problems grows, the polynomial

Table 3 CCCCP: test instance $|P| = 43$, $|N| = 10$, arc density 10 %

K	L	SPLIT-MTZ			AND-L			EE-MTZ		
		Time	MaxWt	MaxMat	Time	MaxWt	MaxMat	Time	MaxWt	MaxMat
3	3	8.22	114	43	13.09	124	43	0.34	128	43
	4	4.87	120	43	17.66	130	43	0.36	128	43
	5	0.66	143	43	2.67	136	43	0.34	124	43
	6	0.25	121	43	8.31	119	43	0.36	127	43
	10	0.36	130	43	2.43	123	43	0.34	119	43
	20	0.25	109	43	7.52	119	43	0.34	124	43
	∞	0.16	119	43	7.46	122	43	0.31	122	43
4	5	4.60	148	43	2.73	136	43	0.92	138	43
	10	2.28	120	43	2.36	123	43	0.78	116	43
	∞	1.08	125	43	2.79	138	43	0.87	124	43
5	5	21.87	125	43	41.57	123	43	1.03	133	43
	10	8.86	126	43	10.55	121	43	1.31	115	43
	∞	4.23	123	43	5.85	112	43	0.87	127	43
6	6	167.95	128	43	405.38	130	43	1.37	128	43
	10	25.34	128	43	86.35	125	43	1.53	134	43
	∞	9.39	123	43	421.87	129	43	1.39	126	43

Objective: maximize total number of matches. Times are in seconds. “MaxWt” and “MaxMat” refer to the values of sum of arc weights and number of matches respectively in the optimal solution

Table 4 More test instances

K	L	SPLIT-MTZ			EE-MTZ		
		Time	MaxWt	MaxMat	Time	MaxWt	MaxMat
$\|V\| = 162$, $\|N\| = 10$, $\|P\| = 152$, arc den 4 %, max $w_{ij} = 10$. Obj: Max Sum Arc Wts							
3	20	1800+	–	–	1800+	–	–
3	∞	14.20	1176	149	43.23	1176	149
6	∞	1800+	–	–	45.86	1177	149
10	∞	1800+	–	–	39.76	1177	149
$\|V\| = 162$, $\|N\| = 10$, $\|P\| = 152$, arc den 4 %, max $w_{ij} = 10$. Obj: Max #Matches							
3	∞	1300.44	801	150	48.47	805	150
6	∞	1800+	–	–	1800+	–	–
10	∞	1800+	–	–	1720.46	856	150
$\|V\| = 203$, $\|N\| = 3$, $\|P\| = 199$, arc den 6 %, max $w_{ij} = 10$. Obj: Max Sum Arc Wts							
3	∞	67.92	1818	199	195.84	1818	199
6	∞	1800+	–	–	211.47	1818	199
10	∞	1800+	–	–	211.91	1818	199
$\|V\| = 256$, $\|N\| = 6$, $\|P\| = 250$, arc den 5 %, max $w_{ij} = 10$. Obj: Max Sum Arc Wts							
3	∞	173.82	2311	250	OOM	–	–

Times are in seconds. “OOM” stands for “out-of-memory”. “MaxWt” and “MaxMat” refer to the values of sum of arc weights and number of matches respectively in the optimal solution. Notice that the objective of some of the test instances below are to maximize sum of arc weights whilst others are to maximize the number of matches. The objectives are indicated in the lines where test instance parameters are described

size formulation failed to even load the model. The size of L does not seem to have any effect on EE-MTZ model. For the SPLIT-MTZ, on the other hand, the larger L is, the faster an optimal solution is obtained. When K increases, however, the problems are in general harder to solve, although, the impact on the EE-MTZ model is not as profound compared to the SPLIT-based models. This is likely due to the fact that currently the SPLIT-MTZ has all cardinality-infeasible-cycle elimination constraints exhaustively generated and included in the ILP, hence the poor performance as K grows. As we have proposed, an Assignment Problem (AP)-based Lagrangean Relaxation within a branch-and-bound framework is expected to improve the performance of the SPLIT-MTZ substantially because AP-relaxations can be solved in polynomial time and that it produces natural integer solutions, hence identifying cardinality-infeasible cycles can simply be done by stepping through the integer solution which should only take a trivial amount of time. As K increases, more subtours induced by the AP-relaxation solution will be cardinality feasible, hence the number of cardinality-infeasible cycle elimination constraints that need to be generated is expected to decrease as K increases.

In handling the chain restrictions, it seems that the MTZ-like constraints proposed in this paper performed more favourably than (39)–(41) proposed in Anderson et al. (2015).

The KEP with the objective of maximizing total number of matches is in general much harder to solve than the objective of maximizing sum of arc weights, which is not surprising as the MILP of the former contains a lot more symmetry by equivalent objective values due to unit arc weights. From the results in the tables, we also observed that when we optimize the sum of arc weights, the value of total number of matches in the optimal solutions are in general close or even equal to the total number of matches when the MILPs are solved with the objective of maximizing total number of matches. The converse is not true—when we maximize the total number of matches, the value of sum of arc weights is usually far from the optimal value when the objective is to maximize sum of arc weights. Perhaps a more efficient way to obtain the optimal solution for the objective of maximizing total number of matches is in fact to maximize the sum of arc weights in the first instance, then apply constraint programming or heuristic approaches to search for an improvement in the total number of matches.

7 Pre-processing

For the CCMcP, in Constantino et al. (2013), a pre-processing algorithm was proposed and tested for the EE model, and can be explained as follows. Observe that the ℓth clone of digraph D can only have cycles that contain the vertex ℓ, and that the only other vertices in these cycles must be those with indices not less than ℓ. One is therefore able to remove some vertices and arcs, and obtain a reduced graph as follows. Let $D^\ell = (V^\ell, A^\ell)$ be the subgraph of the ℓth clone of D, and let d^ℓ_{ij} be the distance of the shortest path on D^ℓ from vertex i to vertex j (in terms of number of arcs between the two vertices). We have that:

$$V^\ell = \left\{ i \in V \mid i \geq \ell,\ d^\ell_{\ell i} + d^\ell_{i\ell} \leq K \right\}$$
$$A^\ell = \left\{ (i,j) \in A \mid i,j \in V^\ell,\ d^\ell_{\ell i} + 1 + d^\ell_{j\ell} \leq K \right\}$$

The pre-processing algorithm is relatively efficient (as only shortest path problems are involved) and appears to be very effective for low density problem instances with $K = 3$ and 4 [up to 97.2 % variables removed, as reported in Constantino et al. (2013)]. Even for high density test instances with $K = 5$ and 6, the reduction in number of vertices/arcs is still impressive, (with around 65 % variables removed).

The pre-processing algorithm described above is not valid for IP models that do not involve cloning D into many copies. We can, however, modify the idea above for graph reduction on D in general. Let d_{ij} be the distance of the shortest path on D from vertex i to vertex j with unit arc weights. Consider:

$$\widetilde{V} = \left\{ j \in P \mid \left(\exists \sigma \in N \text{ s.t. } d_{\sigma j} \leq L\right) \vee \left(\exists i \in P \text{ s.t. } d_{ij} \leq K-1\right)\right\}$$
$$\hat{A} = \left\{ (i,j) \in A \mid i,j \in \widetilde{V},\ \exists \sigma \in N \text{ s.t. } d_{\sigma i} + 1 + d_{j\tau} \leq L+1 \right\}$$
$$\widetilde{A} = \left\{ (i,j) \in A \mid i,j \in \widetilde{V},\ d_{ji} \leq K-1 \right\}$$

For arc-based model such as Model 4 for the CCMcP and Model 6 for the CCCCP, we can remove all cycle-related arc variables (x_{ij} and u_{ij} respectively) for $(i,j) \in A \setminus \widetilde{A}$. For Model 6, we can also remove all chain-related arc variables (y_{ij}) for $(i,j) \in A' \setminus \hat{A}$. We cannot do so, however, for Model 5 for the CCCCP, as the same variable is used to indicate an arc as being part of a cycle or a chain.

8 Multiple objectives

In Glorie et al. (2014), a set of "hierarchical" objectives used in the Dutch National Kidney Exchange Program were described. In the first instance, the objective is to maximize the *number of transplants*, i.e., $z_1 = \max \sum_{(i,j)\in A} w_{ij}x_{ij}$, with the weight of arcs w_{ij} set to 1 for all $(i, j) \in A$. Let z_1^* be the optimal value. It was proposed that other objectives can be optimized in a lexicographical fashion. For example, in the second instance, the objective is to maximise the *number of blood type identical transplants*. Let $B \subseteq A$ be the set of arcs that represent blood type identical transplants, the objective function will now be $z_2 = \max \sum_{(i,j)\in B} w_{ij}x_{ij}$, subject to an additional constraint that $\sum_{(i,j)\in A} w_{ij}x_{ij} \geq z_1^*$, in other words, to perform an objective propagation.

The next four objectives include "match the patients in priority order based on match probability", "minimize the length of the longest cycle or chain", "maximise the spread over transplant centres per cycle and chain", and "match the patient with the longest waiting time". Although the idea of iterative objective propagation was proposed, there were no mathematical description on how exactly these are modelled, in particular for the second last objective. Only the first objective, i.e., to maximize the number of matches, was included in their numerical experiments.

In Manlove and O'Malley (2012), a different set of objectives were described. They are mostly to do with promoting cycles with smaller sizes to be used. A 2-cycle is considered more favourable than a 3-cycles, unless there exists one or more 2-cycles embedded in a 3-cycle. A lexicographical approach similar to that of Glorie et al. (2014) was described, but again, there is no numerical results to demonstrate the respective computation times for each round of the objective propagation.

In our experience with integer programming, after solving the first ILP for the first-priority objective function, if we solve a second ILP with the second-priority objective function while having the first objective propagated as a constraint, even though the feasible space is much smaller, it does not always guarantee the second IP can be solved much faster. In fact, sometimes it can take much longer. In Table 5, the columns labelled "MaxMat" are the computation times required for maximizing the number of matches. These times are extracted from Table 3. The columns labelled "MaxWt" are the computation times required for solving the second ILP (with an objective of maximizing sum of arc weights), given the first ILP already solved to optimality and the optimal number of matches propagated as a constraint in the second ILP. We can clearly see that the computation can be substantially longer in many cases. One significant contribution for future research is to properly address the multi-criteria nature of the Kidney Exchange Program.

9 Future research directions

In Mak and Boland (2007), it was demonstrated that an AP-based Lagrangean Relaxation within a BNB framework performed more favourably than the branch-and-cut-and-price method of Boland et al. (2000) in the context of RATSP. For the CCMcP, when the cycle-cardinality constraints are relaxed, the resulting problem is an

Table 5 Test instance $|P| = 43$, $|N| = 10$, arc density 10 %

K	L	SPLIT-MTZ time(s)		And-L time(s)		EE-MTZ time(s)		
		MaxWt	MaxMat	MaxWt	MaxMat	MaxWt	MaxMat	Opt
3	3	1200+	8.22	1200+	13.09	1200+	0.34	–
	4	1200+	4.87	1200+	17.66	583.32	0.36	164
	5	42.96	0.66	1200+	2.67	13.68	0.34	172
	6	3.03	0.25	11.81	8.31	1.53	0.36	174
	10	0.36	0.36	0.48	2.43	0.47	0.34	178
	20	0.14	0.25	0.56	7.52	0.44	0.34	178
	∞	0.42	0.16	0.45	7.46	0.42	0.31	178
4	5	102.18	4.60	190.01	2.73	23.42	0.92	172
	10	0.47	2.28	2.29	2.36	0.67	0.78	178
	∞	0.51	1.08	1.98	2.79	0.58	0.87	178
5	5	97.27	21.87	727.65	41.57	33.20	1.03	174
	10	2.07	8.86	8.24	10.55	0.80	1.31	178
	∞	1.75	4.23	4.49	5.85	0.81	0.87	178
6	6	161.41	167.95	594.41	405.38	3.04	1.37	176
	10	10.28	25.34	18.44	86.35	0.78	1.53	178
	∞	11.29	9.39	16.22	421.87	0.78	1.39	178

Objective: maximize sum of arc weights subject to number of matches propagated from the optimal objective value obtained from experiments in Table 3. The computation times are recorded in the columns "MaxMat" and "MaxWt" respectively. Arc weight w_a randomly generated from $\{1, \ldots, 5\}$. The column entitled "Opt" is the value of the maximum sum of arc weights (for comparisons with the MaxWt values in Table 3)

AP. Hence one promising research direction is to implement a similar methodology. For the CCCCP, whilst the polynomial-sized SPLIT model, (Model 2), appears to be computationally superior for problems of small to moderate sizes, for larger problems, due to its $\mathcal{O}(|V|^3)$ number of variables, the exponential-sized SPLIT model, (Model 6), is more promising in handling larger problems, particularly in a cut-generation manner. More research can be conducted in the efficient and effective application of the exponential-sized SPLIT model.

Another research direction is to properly address the multi-criteria nature of the KEP. In existing literature, objective propagation within an integer programming framework has been mentioned in Manlove and O'Malley (2012) and proposed in Glorie et al. (2014). No computational details can be found in the literature. As demonstrated in Table 5, despite dealing with a much smaller feasible set, it is not always true that the subsequent ILPs obtained by objective propagation can be solved efficiently. Heuristic approaches, Constraint Programming (CP), and a hybrid of ILP with heuristic or CP approaches for multi-criteria optimization of the KEP can be explored for future research.

Acknowledgements To "The α and the ω".

References

Abraham DJ, Blum A, Sandholm T (2007) Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. In: Proceedings of the 8th ACM conference on electronic commerce, EC '07. ACM, New York, pp 295–304

Anderson R, Ashlagi I, Gamarnik D (2015) Finding long chains in kidney exchange using the traveling salesman problem. Proc Natl Acad Sci 112:663–668

Ashlagi I, Gilchrist DS, Roth AE, Rees MA (2011) Nonsimultaneous chains and dominos in kidney- paired donation revisited. Am J Transpl 11:984–994

Bai G (2009) A new algorithm for k-cardinality assignment problem. In: International conference on computational intelligence and software engineering, 2009, CiSE 2009, pp 1–4

Baldacci R, Toth P, Vigo D (2010) Exact algorithms for routing problems under vehicle capacity constraints. Ann Oper Res 175:213–245

Bauer P, Linderoth J, Savelsbergh M (2002) A branch and cut approach to the cardinality constrained circuit problem. Math Program 91:307–348

Biró P, Manlove D, Rizzi R (2009) Maximum weight cycle packing in directed graphs, with application to kidney exchange programs. Discret Math Algorithms Appl 1:499–517

Boland N, Clarke L, Nemhauser G (2000) The asymmetric traveling salesman problem with replenishment arcs. Eur J Oper Res 123:408–427

Cao B, Glover F (1997) Tabu search and ejection chains&mdash;application to a node weighted version of the cardinality-constrained tsp. Manage Sci 43:908–921

Chen Y, Kalbfleisch JD, Li Y, Song PXK, Zhou Y (2012) Computerized platform for optimal organ allocations in kidney exchanges

Constantino M, Klimentova X, Viana A, Rais A (2013) New insights on integer-programming models for the kidney exchange problem. Eur J Oper Res 231:57–68

Cornuejols G, Harche F (1993) Polyhedral study of the capacitated vehicle routing problem. Math Program 60:21–52

Dell'Amico M, Martello S (1997) The k-cardinality assignment problem. Discret Appl Math 76:103–121

Dickerson JP, Procaccia AD, Sandholm T (2012) Optimizing kidney exchange with transplant chains: theory and reality. In: Proceedings of the 11th international conference on autonomous agents and multiagent systems, vol. 2, AAMAS '12. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, pp 711–718

Fischetti M, Gonzlez JJS, Toth P (1998) Solving the orienteering problem through branch-and-cut. INFORMS J Comput 10:133–148

Gate SF (2015) 9-way kidney swap involving 18 surgeries at 2 S.F. hospitals. http://www.sfgate.com/health/article/9-way-kidney-swap-involving-18-surgeries-at-2-6307975.php

Gentry SE, Segev DL, Simmerling M, Montgomery RA (2007) Expanding kidney paired donation through participation by compatible pairs. Am J Transpl 7:2361–2370

Gentry SE, Montgomery RA, Swihart BJ, Segev DL (2009) The roles of dominos and nonsimultaneous chains in kidney paired donation. Am J Transpl 9:1330–1336

Gentry SE, Montgomery RA, Segev DL (2011) Kidney paired donation: fundamentals, limitations, and expansions. Am J Kidney Dis 57:144–151

Glorie KM, van de Klundert JJ, Wagelmans APM (2014) Kidney exchange with long chains: An efficient pricing algorithm for clearing barter exchanges with branch-and-price. Manuf Serv Oper Manage 16:498–512

Hartmann M, Özlük Ö (2001) Facets of the p-cycle polytope. Discret Appl Math 112:147–178. Combinatorial Optimization Symposium, Selected Papers

Kaibel V, Stephan R (2007) On cardinality constrained cycle and path polytopes. http://arxiv.org/pdf/0710.3036v1.pdf

Kaibel V, Stephan R (2010) On cardinality constrained cycle and path polytopes. Math Program 123:371–394

Kidney Health Australia (2015). http://www.kidney.org.au/KidneyDisease/FastFactsonCKD/tabid/589/Default.aspx

Klimentova X, Alvelos F, Viana A (2014) A new branch-and-price approach for the kidney exchange problem. In: Murgante B et al (eds) Computational science and its applications–ICCSA 2014. Lecture notes in computer science, vol 8580. Springer, pp. 237–252

Mak V, Boland N (2000) Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs. Int Trans Oper Res 7:431–447

Mak V, Boland N (2006) Facets of the polytope of the asymmetric travelling salesman problem with replenishment arcs. Discret Optim 3:33–49

Mak V, Boland N (2007) Polyhedral results and exact algorithms for the asymmetric travelling salesman problem with replenishment arcs. Discret Appl Math 155:2093–2110

Mak-Hau V (2015) Polyhedral results for the cardinality constrained multi-cycle problem (CCMcP) and the cardinality constrained cycles and chains problem (CCCCP). http://www.deakin.edu.au/~vicky/TechnicalReport2.pdf

Manlove D, O'Malley G (2012) Paired and altruistic kidney donation in the UK: algorithms and experimentation. In: Klasing R (ed) Experimental algorithms. Lecture notes in computer science, vol 7276. Springer, Berlin, pp 271–282

Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesman problems. J ACM 7:326–329

Nguyen VH, Maurras J (2001) On the linear description of the *k*-cycle polytope. Int Trans Oper Res 8:673–692

Patterson R, Rolland E (2003) The cardinality constrained covering traveling salesman problem. Comput Oper Res 30:97–116

Roth AE, Sünmez T, Ünver MU (2007) Efficient kidney exchange: coincidence of wants in markets with compatibility-based preferences. Am Econ Rev 97:828–851

Saidman SL, Roth AE, Sönmez T, Ünver MU, Delmonico FL (2006) Increasing the opportunity of live kidney donation by matching for two and three way exchanges. Transplantation 81:773–782

Toth P, Vigo D (2002) Models, relaxations and exact approaches for the capacitated vehicle routing problem. Discret Appl Math 123:487–512

Zenios SA, Chertow GM, Wein LM (2000) Dynamic allocation of kidneys to candidates on the transplant waiting list. Oper Res 48:549–569