



## **Using Optimisation Techniques on the Kidney Exchange Problem**

Final Report

CM3203 – One Semester Individual Project - 40 Credits

Author: Molly Wilson

Supervisor: Dr Richard Booth

Moderator: Dr Bailin Deng

School of Computer Science and Informatics

Cardiff University

14<sup>th</sup> May 2021

## **Abstract**

Every year, there are on average 4,536 patients on the active waiting list for a kidney transplant in the UK. Research is ongoing to develop effective ways to allocate donations to save the lives of patients and maintain their quality of life. This project aims to apply optimisation techniques to datasets modelling the kidney exchange and analyse the effects of different methods and definitions of optimality. Four main approaches are explored as part of this project, including Pairwise Exchange, Integer Linear Programming, Top Trading Cycles and Chains, and introducing matching patient-donor pairs into the matching pool. Analysis of the varying methods and datasets indicated that the Integer Linear Programming formulation is the method which provides solutions with the highest cardinality. The results also demonstrate that the inclusion of compatible patient-donor pairs to the matching pool has a significant effect on the number of patients allocated a kidney as part of the solution. Further research is required to apply these findings to an international exchange model.

## **Acknowledgements**

I would like to thank my supervisor, Dr Richard Booth, for his support and guidance throughout the project, for which I am grateful. In addition, I am grateful for his lectures on Combinatorial Optimisation, which developed my interest in optimisation and inspired my choice of project.

## Table of Contents

Abstract .....	2
Acknowledgements .....	3
1 Introduction .....	6
2 Background .....	8
2.1 The Matching Process .....	8
2.2 Kidney Exchange Program .....	9
2.3 Similar Research .....	10
3 Implementation of Algorithms .....	12
3.1 Mathematical Description of the Problem .....	12
3.2 Optimisation .....	13
3.3 Dataset .....	13
3.4 Pairwise Exchanges .....	13
3.4.1 Pairwise Exchange Mathematical Description .....	14
3.4.2 Edmond's Blossom Algorithm .....	15
3.4.3 Example .....	15
3.4.4 Pseudocode .....	17
3.4.5 Implementation .....	19
3.5 Integer Linear Programming .....	20
3.5.1 Kosaraju's Algorithm .....	20
3.5.2 Kosaraju's Algorithm Steps .....	21
3.5.3 Kosaraju's Algorithm Pseudocode .....	21
3.5.4 Example .....	22
3.5.5 Johnson's Algorithm .....	23
3.5.6 Johnson's Algorithm Steps .....	24
3.5.7 Johnson's Algorithm Pseudocode .....	24
3.5.8 Example .....	26
3.5.9 Implementation .....	26
3.5.10 ILP Formulation .....	27
3.5.11 Example .....	28
3.5.12 Limitations .....	29
3.6 Top Trading Cycle .....	30
3.6.1 Top Trading Cycle Description .....	30
3.6.2 Pseudocode .....	31

3.6.3 Example .....	32
3.6.4 Extending the Dataset .....	34
3.6.5 Implementation .....	35
3.7 Top Trading Cycle and Chains .....	36
3.7.1 Example .....	37
4 Results and Evaluation.....	38
4.1.1 Dataset 1.....	38
4.1.2 Dataset 2.....	39
4.1.3 Dataset 3.....	40
4.2.1 Pairwise Exchange Results Dataset 1 .....	41
4.2.2 Pairwise Exchange Results Dataset 2 .....	42
4.2.3 Pairwise Exchange Results Dataset 3 .....	44
4.3.1 ILP Results Dataset 1 .....	46
4.3.2 ILP Results Dataset 2.....	51
4.3.3 ILP Results Dataset 3.....	57
4.4.1 Top Trading Cycle Results Dataset 1 .....	62
4.4.2 Top Trading Cycle Results Dataset 2 .....	64
4.3.3 Top Trading Cycle Results Dataset 3 .....	68
5 Future Work .....	75
6 Conclusions .....	76
7 Reflection on Learning.....	78
Table of Abbreviations.....	81
References .....	82

## Table of Tables

Table 1: Sample of Graph Data for Dataset $D_1$ .....	38
Table 2: Patient-Donor Details for Pairs in Dataset $D_1$ .....	39
Table 3: Sample of Additional Edges in Dataset $D_2$ .....	40
Table 4: Altruistic Donor Information for Dataset $D_2$ .....	40
Table 5: Compatible Patient-Donor Pair Information for Dataset $D_3$ .....	40
Table 6: Compatible Patient-Donor Matching Information for Dataset $D_3$ .....	41
Table 7: Donor Preferences for each Patient in Dataset $D_1$ .....	62
Table 8: Donor Preferences for each Patient in Dataset $D_2$ .....	65
Table 9: Donor Preferences for each Patient in Dataset $D_3$ .....	68
Table 10: Results Table for Dataset $D_1$ .....	71
Table 11: Results Table for Dataset $D_2$ .....	72
Table 12: Results Table for Dataset $D_3$ .....	73

# 1 Introduction

Chronic Kidney Disease (CKD) is a long-term health condition whereby kidney function deteriorates gradually over a period of months or years. There are 1.8 million people living with CKD in the UK, with an additional one million estimated cases of undiagnosed CKD [8]. As of 29th February 2020, there were 4,726 patients on the UK active kidney transplant list, of which 108 were paediatric patients. This represents a 1% decrease in adult patients from the previous year and a 17% increase in paediatric patients [6]. In addition, 3,190 adult kidney transplants were performed in the UK in 2019/20, indicating a decrease of 3% on the previous year. Of these, 1,326 were from donations by brainstem death donors (DBD), 915 were from donation by cardiac death donors (DCD) and 949 were from living donors [6].

Left untreated, CKD leads to kidney failure, and dialysis or transplant is required to prevent death. It is therefore, of utmost importance that kidney donation and allocation is managed effectively to save lives. Currently, the median waiting time for an adult kidney transplant is between 2 and 3 years and in 2020, 233 people passed away whilst awaiting a kidney transplant, which was more than the number of patients who died awaiting lung, liver and heart transplants combined [19]. Currently, the level of kidney donation in the UK is not sufficient to meet the demand for transplants. It is thus of paramount importance that kidney exchanges and transplants are managed effectively to prolong patient quality of life.

The UK Living Kidney Sharing Scheme (UKLKSS) facilitates transplants for incompatible patient-donor pairs by introducing them into the matching pool. In the matching pool, the donor is allocated to a compatible patient in need of a transplant on the promise that the patient they entered the pool with is also allocated a kidney from one of the other donors. There are several formats and shapes in which a matching occurs from the pool such as a pairwise exchange, a cycle exchange or an altruistic domino chain (ADC), each of which will be explained and explored within this project.

The aim of this project is to identify and explore appropriate optimisation techniques to model the kidney exchange, implement the identified techniques and produce results for each method in the process of kidney allocations. Varying datasets will be utilised to demonstrate the performance of each algorithm and the resulting optimal solutions will be analysed. Finally, from these methods, the most appropriate and successful techniques will be identified for modelling the kidney exchange.

This project will also explore differing approaches to solving optimisation problems such as Integer Linear Programming, and algorithms such as the Top Trading Cycle and Edmond's Blossom Algorithm to compare and contrast their effects on the application of the kidney exchange. The project will also explore several definitions of optimality, such as maximum cardinality (which shows maximum patients matched), and maximum weight (which shows the probability of a successful transplant). Finally, the project explores the impact of altruistic donors and the introduction of compatible patient-donor pairs into the matching pool on the cardinality of maximum matching and utility of available donors.

This project begins with an overview of kidney disease and its devastating impacts, then a brief history of the evolution of the kidney exchange will be considered, primarily focussing on progress in the UK, and current suggested methods to optimise kidney allocation. Following this, Chapter three offers an overview of each of the four approaches included in this project: Integer Linear Programming, Top Trading Cycle and Chains, Pairwise Exchanges, and the Introduction of Matching Patient-Donor Pairs into Matching Pools. Each of the approaches will be implemented and analysed in-depth before the results are collated and critically evaluated in Chapter four. To conclude this project, Chapters five and six will offer suggestions for future work and a concluding argument. Chapter seven presents a reflection of my learning and progress throughout the project.

## 2 Background

### 2.1 The Matching Process

There are four different blood types O, A, B and AB. Patients with blood type O, the most common blood type, can only be matched with donors who are also of blood type O. Patients with blood type A or type B can be matched with donors of the same blood type and blood type O. Patients with blood type AB can be matched with donors of any blood type. For this reason, patients with blood type AB are considered universal recipients. As donors of blood type O can be matched with patients of any blood type, they are considered universal donors [7].

There are a large number of factors considered when evaluating the compatibility between a potential patient-donor match. Some of these include the age difference between the patient and donor, the sensitivity level, the HLA mismatch level, and previous matching run points [6].

The following outlines the structure for defining the compatibility of a match within the NHS:

1. Previous Matching Run Points. This factor is measured by multiplying the number of previous matchings the patient has already taken part in by a factor of fifty. The current matching algorithm is executed quarterly in January, April, July, and October.
2. Sensitisation Points. Sensitivity of a patient is measured by a Panel Reactive Antibody Test. This test determines the number of pre-existing antibodies the patient has against human cell antigens. The result of this test is a percentage which represents the proportion of the population with antigens to which the patient will react to. This factor is represented by dividing the sensitivity percentage by 2.
3. Estimated Human-Leukocyte Antigens (HLA) Mismatch. HLA are proteins found in tissue and on the surface of white blood cells. Where a patient and donor share the same HLA, this is referred to as a tissue-type match, meaning that their tissue is immunologically compatible. The number of compatible proteins between a patient and donor is assigned a value of 0, 5, 10 or 15, where the lowest values represent a high level of incompatibility and the highest values represent a low level of incompatibility.



4. Donor Age Difference. Where the age gap between the donor and the patient is less than 20 years, they are assigned a value of 3. Where this gap is greater than 20 years, it is assigned a value of 0. The age difference between the patient and the donor is minimised as a priority due to increased risk of graft failure as the difference increases. For example, the acute rejection rate rises from 10.7% to 32% when the age difference between the donor and patient is greater than 20 years [10].

The sum of the values from each of these categories represents the level of compatibility of a patient-donor match, where high values represent a greater match.

## **2.2 Kidney Exchange Program**

The UKLKSS began in January 2012 and is the collective term for the various methods in which kidneys are donated and transplanted across the UK. It was established following the introduction of the Human Tissue Act 2004 and has become one of the most effective schemes in Europe [14]. Before the introduction of this legislation, it was not possible for transplants to take place between a patient and donor who did not have a genetic or emotional connection. As a result, patient-donor pairs who were not compatible with one another were rejected for transplant, the willing donor sent home and the patient left to join the kidney transplant waiting list. Evidently, this was a waste of resources as the willing donor was unable to participate in a transplant and consequently their paired patient, and other patients who are compatible with the donor were impacted as a result. The creation of the kidney exchange program aims to utilise the opportunity of a willing donor to allocate a kidney to a patient in need, on the promise that the incompatible paired patient will also receive a kidney. Methods encompassed by the UKLKSS include paired/pooled donation (PPD) and ADCs initiated by non-directed altruistic donors (NDADs).

The kidney exchange allows patient-donor pairs who are not compatible with one another to participate in a matching pool where the donor is matched with another patient in need of a transplant and the patient receives a kidney from another patient's incompatible donor. There are several variations of which a matching may occur in this situation. One circumstance, called a two-way or pairwise matching, occurs when two patient-donor pairs  $p_1d_1$  and  $p_2d_2$  match such that patient  $p_1$  receives the kidney of donor  $d_2$  and patient  $p_2$  receives the kidney of donor  $d_1$ . It is possible for paired exchanges to occur with more than two patients such that the solution forms a cycle. This means that for patient-donor pairs  $p_1d_1$ ,  $p_2d_2$  and  $p_3d_3$  form a matching

such that patient  $p_1$  receives the kidney of donor  $d_2$ , patient  $p_2$  receives the kidney of donor  $d_3$  and patient  $p_3$  receives the kidney of donor  $d_1$ . This forms a cycle such that each donor is able to donate their kidney to a patient who requires it, and each patient is able to receive a kidney. In theory, there is no limit to the number of pairs who may be included in this cycle. However, in practice, each transplant must be performed simultaneously. This is due to the fact the donors are not legally obliged to donate their kidney, such that it is possible for a donor to leave the program before their donation [11]. As a result, the number of pairs  $k$  in any cycle is kept small, usually between three and five. A third way a match may occur is when a non-directed anonymous donor, also known as an altruist, triggers a chain of kidney donations. This is referred to as an altruistic domino chain.

According to NHS Blood and Transplant, the UKLKSS allowed 71 extra patients to receive a kidney transplant from a living donor in 2019 compared with 2018 [20].

### **2.3 Similar Research**

Manlove and O'Malley (2014) carried out experiments using genuine data to analyse the number of transplants which actually took place within in each quarter, compared to results generated by two different ILP formulations and experiments on constraints such as cycle length. One conclusion that arises from this study is that the introduction of four-way exchanges into the matching pool is likely to lead to a considerable increase in kidney transplants. The conclusions drawn in this paper have inspired the ideas presented as part of this project, as one of the methods this project explores is also an ILP formulation. [12]

In 2009, Biró, Péter & Manlove (2009) modelled the kidney exchange as a packing problem and proved the approximable-completeness of finding a maximum cardinality solution using only 2-cycles and 3-cycles. The resulting exact algorithm produced by this research provides optimal solutions for the National Matching Scheme for Paired Donation run by NHS Blood and Transplant. An approximation algorithm was also produced to find a maximum weight solution containing cycles of bounded length. [1]

Recent published research into the kidney exchange states that “single best practise models and methods for Kidney Exchange Programmes do not exist”. This statement demonstrates the importance of continued research into different methods and circumstances. Optimal solutions differ depending on the ratio of donors to patients, ability to perform successful transplants simultaneously and the number of donations

available from each type of donor, such as non-directed altruistic and DCD. This research also ventures the concept of cross-national kidney exchange programmes. Inspired by these models, this project explores the effect of introducing compatible patient-donor pairs into the matching pool. This data mirrors the situation where sharing waiting list information internationally could facilitate additional transplants when compared with allocating kidneys nationally. [2]

### 3 Implementation of Algorithms

#### 3.1 Mathematical Description of the Problem

Consider a directed graph  $G(n)$  such that  $n$  equals the number of patients awaiting a kidney in this exchange. For each of the  $n$  donor-patient pairs, there exists a node in the graph  $G(n)$ . Where there is a match between the donor kidney  $n_i$  and patient  $n_j$ , add an edge  $e$  between node  $n_i$  and  $n_j$ . The weight  $w_e$  of the edge  $e$  is equal to the utility of  $n_j$  receiving  $n_i$ 's donor kidney. Please note, a donor in a pair is willing to donate their kidney if and only if their paired patient receives one.

A cycle  $c$  in the graph represents a possible exchange, with each node in the cycle receiving the kidney of the previous node. The weight  $w_c$  of a cycle  $c$  is the sum of its edge weights. A complete matching consists of disjoint cycles, as each donor can only donate one of their kidneys. Each cycle  $c$  must be of length  $k$  where  $2 \leq k \leq 5$ , and in most practical cases  $k = 3$ .  $k$  must be small as all operations in the cycle are performed simultaneously.

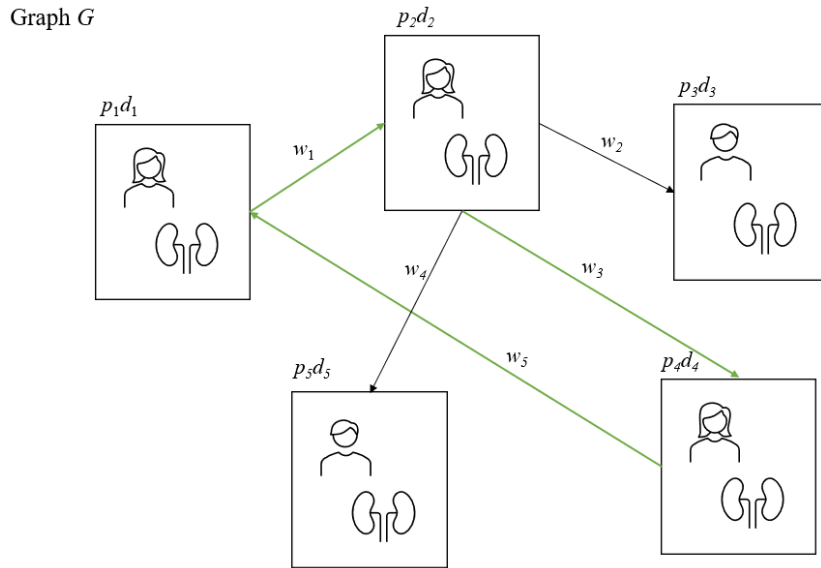


Figure 1: Basic Graph  $G$

In Figure 1 above, the graph  $G$  contains five patient-donor pairs, with five edges which represent the matches between patients and donors. A cycle of length 3 is indicated by the green arrows.

In the case of an altruistic donor, the node  $n$  will represent the donor. The donor is prepared to donate unconditionally, meaning that there is no incoming edge to the node  $n$ . In this case, a complete matching is not represented by a cycle  $c$  but a chain known as an altruistic donor chain.

An optimal matching in this problem maximises the weight of cycles and chains, meaning the number of successful transplants is also maximised with the available resources.

### **3.2 Optimisation**

Within this problem, there are varying definitions of what constitutes an optimal solution due to the nature of transplants and their effect on a patient's quality of life or chance of survival. As a result, there are a range of factors to consider when defining optimality. This includes, but is not limited to, maximising the sum of the matching scores in the solution. This gives weight to factors such as length of time on the waiting list, difficulty of matching the patient, and maximising the number of patients in the solution. In addition, priority may be awarded to patients who are most unwell or to younger patients as they are more likely to benefit long-term from a transplant. There are obvious ethical issues surrounding priority amongst patients and ultimately the decision lies with the founders of each individual transplant scheme to define a solution within their scheme. Within the NHS, an optimal solution is defined as first prioritising obtaining a maximum matching score and then maximising the number of patients included in the solution.

### **3.3 Dataset**

The datasets used throughout the project are provided by [preflib.org](http://preflib.org). Although the datasets are synthetic, it was produced by a state-of-the-art donor pool generation method, described by Saidman (2006) [16]. Each dataset consists of two files, one to document the patient-donor pair information, and another containing information regarding patient-donor compatibility, which represent the edges in the associated graph.

### **3.4 Pairwise Exchanges**

Prior to sufficient research conducted into the theory and practicalities of 3-cycle solutions or greater, the kidney exchange problem existed in a simplified state known as a pairwise exchange. This means that an optimal solution consisted of patient-donor pairs who had been successfully matched to one another. For example, for three patient-donor pairs  $p_1d_1$ ,  $p_2d_2$  and  $p_3d_3$ , the maximum cardinality of the optimal

solution is two. This solution may be such that patient  $p_1$  receives a kidney from donor  $d_2$  and patient  $p_2$  receives a kidney from donor  $d_1$ .

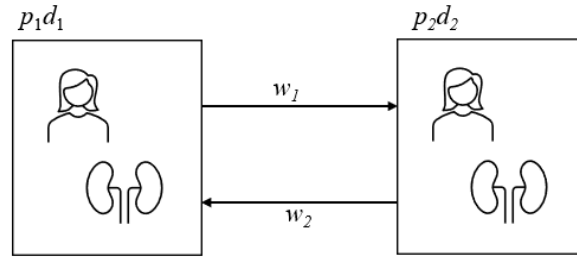


Figure 2: Pairwise Exchanges

The primary focus in the NHS is to find a matching that allows the maximum number of patients to be assigned a kidney. This means that maximum cardinality is the priority.

### 3.4.1 Pairwise Exchange Mathematical Description

For each patient-donor in the matching pool, there exists a vertex  $p_x d_x$  in the directed graph  $D$ .

For each match between a patient  $p_x$  and a donor  $d_y$ , there exists a directed edge from vertex  $p_x d_x$  to vertex  $p_y d_y$  where the weight of the edge is equal to the matching score for the transplant from donor  $d_y$  to patient  $p_x$ .

In the case of pairwise exchanges, a directed edge between a vertex  $p_x d_x$  and a vertex  $p_y d_y$  where there does not exist a directed edge in the opposite direction, from vertex  $p_y d_y$  to vertex  $p_x d_x$  is excluded as this edge is not valid in a solution. As a result, the directed graph  $D$  is transformed to an undirected graph  $G$ .

For each vertex  $p_x d_x$  that exists in the directed graph  $D$ , there exists an equivalent vertex  $p_x d_x$  in undirected graph  $G$ .

For each pair of vertices,  $p_x d_x$  and  $p_y d_y$ , connected by two edges,  $e_x$  and  $e_y$ , in the directed graph  $D$ , there exists a single undirected edge  $e_z$  in graph  $G$  between  $p_x d_x$  and  $p_y d_y$  with a weight  $w_z$  where  $w_z = e_x(w_x) + e_y(w_y)$ .

For each pair of vertices,  $p_x d_x$  and  $p_y d_y$ , connected by a single directed edge in graph  $D$ , there does not exist an edge between them in graph  $G$ .

Now, for the undirected graph  $G$ , find a maximum cardinality, maximum weight matching.

### 3.4.2 Edmond's Blossom Algorithm

The solution to this problem is solved by Edmonds's Blossom Algorithm, presented by Jack Edmonds (1965) which is designed to compute a maximum matching in an undirected general graph [3]. It is inspired by and an extension of the Hopcroft-Karp algorithm which computes a maximum matching for bipartite graphs only.

The Blossom Algorithm performs by building augmenting paths starting and ending at unmatched vertices from a graph, alternating between matched and unmatched edges. This process repeats until no more augmenting paths can be identified, meaning that a solution has been found. The algorithm takes its name from the process in which an odd length cycle is identified as this is referred to as a blossom. The nodes which belong to a blossom are contracted, and treated as a single node, such that the graph contains an even number of nodes and is therefore suitable for the Hungarian algorithm, which computes maximum-weight matching in bipartite graphs, to identify further augmenting paths. In the worst case, this Edmond's Blossom Algorithm has a time complexity of  $O(|E||V|^2)$  where  $E$  is the number of edges in the graph and  $V$  is the number of vertices [13].

### 3.4.3 Example

For example, take the graph shown below in Figure 3.

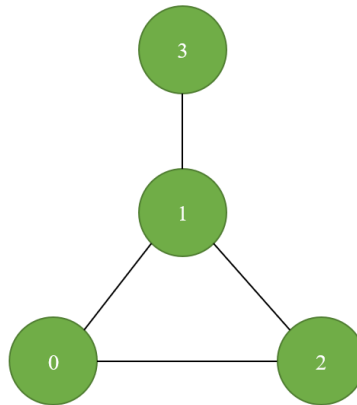


Figure 3: Blossom Algorithm Example Initial Graph

At the start of the algorithm, each of the vertices are unmatched. Starting with vertex  $v_0$ , the algorithm uses breadth first search (BFS) to find a path. The first neighbour of vertex  $v_0$  is  $v_1$ . As vertex  $v_1$  is unmatched, there exists an augmenting path between  $v_0$  and  $v_1$ . After a path is found, all vertices are inverted from matched to unmatched and vice versa. The number of edges included in the matching increases by 1. Figure 4 shows the current path.

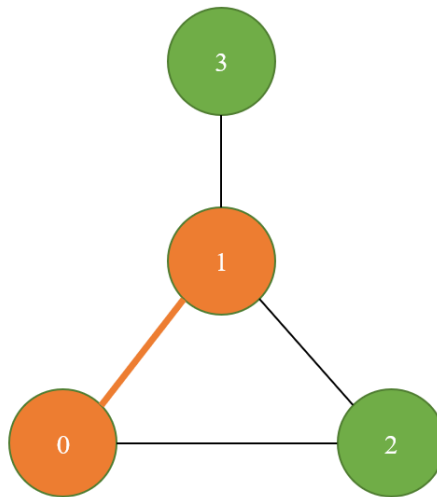


Figure 4: Blossom Algorithm Augmenting Path

There are still two unmatched vertices in the graph  $v_2$  and  $v_3$ . Now starting with  $v_3$ , use BFS to find another path. The first neighbour of  $v_2$  is  $v_0$ , a matched node. As  $v_0$  is connected to  $v_1$ , both nodes are added to the BFS queue. The next neighbour of  $v_2$  is  $v_1$ , which already exists in the queue. Therefore, a cycle has been identified and as it has an odd number of edges, it is called a blossom. The nodes in the blossom are shrunk and represented as one node.



Figure 5: Shrunk Blossom Node

The new blossom node is also added to the BFS queue. Now, BFS is continued starting from the blossom node  $v_4$ . The first neighbour of  $v_4$  is  $v_3$ , an unmatched node. Therefore, an augmenting path has been identified. As there are no more unmatched nodes in the graph, the blossom node is expanded such that the resulting path can be identified.



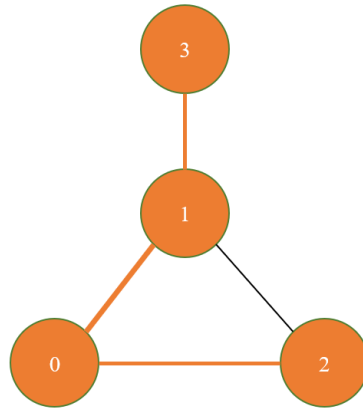


Figure 6: Final Augmenting Path

Now, the matched edges are inverted such that our previous path (0, 1) becomes (2, 0) and (1, 3). As there are no more free vertices, a maximum cardinality matching has been found.

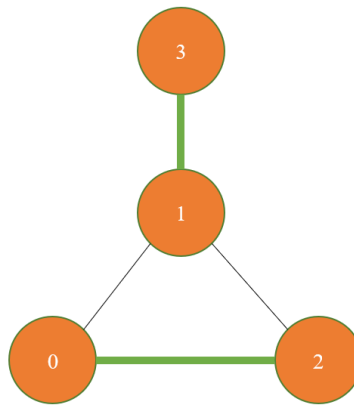


Figure 7: Blossom Algorithm Maximum Cardinality Matching

### 3.4.4 Pseudocode

The following pseudocode is provided by Shoemaker and Vare (2016) [18].

Algorithm 1 Blossom Algorithm: Find Maximum Matching

```

procedure seq_find_maximum_matching( $G, M$ )
     $P = \text{seq\_find\_aug\_path}(G, M)$ 
    if  $P == []$  then
        return  $M$ 
    else
        Add alternating edges of  $P$  to  $M$ 
    return seq_find_maximum_matching( $G, M$ )

```

Algorithm 2 Sequential Blossom Algorithm: Find Augmenting Path

```
procedure seq_find_aug_path( $G, M$ )

 $F$  = empty forest

nodes_to_check  $\leftarrow$  exposed vertices in  $G$ 

for  $v$  in nodes_to_check do

    Add  $v$  as single-node tree to  $F$ 

    node_to_root( $v$ ) =  $v$ 

in  $G$ , mark all matched edges (all edges in  $M$ )

for  $v$  in forest_nodes do

    while there exists an unmarked edge  $e = (v, w)$  do

        if  $w \notin F$  then (Vertex  $w$  must be in  $M$ )

            seq_add_to_forest( $M, F, v, w$ )

        else

            if dist( $w$ , node_to_root( $w$ )) % 2 == 0 then

                if node_to_root( $v$ )  $\neq$  node_to_root( $w$ ) then

                     $P$  = seq_return_aug_path( $F, v, w$ , node_to_root)

                else

                     $P$  = seq_blossom_recursion( $G, M, F, v, w$ )

                return  $P$ 

            else

                # Do nothing

    mark edge  $e$ 

return empty path
```

Algorithm 3 Sequential Blossom Algorithm: Add to Forest

```
procedure seq_add_to_forest( $M, F, v, w$ )

 $x \leftarrow$  vertex adjacent to  $w$  in  $M$ 

add edges  $(v, w), (w, x)$  to tree( $v$ ) in  $F$ 

add vertex  $x$  to nodes_to_check
```

```

node_to_root( $w$ ) = node_to_root( $v$ )

node_to_root( $x$ ) = node_to_root( $v$ )

```

Algorithm 4 Sequential Blossom Algorithm: Return Aug Path

```

procedure seq_return_aug_path( $F$ ,  $v$ ,  $w$ ,  $node\_to\_root$ )

    root_v = node_to_root( $v$ )

    root_w = node_to_root( $w$ )

     $P1 \leftarrow shortest\_path(F, root\_v, v)$ 

     $P2 \leftarrow shortest\_path(F, w, root\_w)$ 

    return  $P1 + P2$ 

```

Algorithm 5 Sequential Blossom Algorithm: Blossom Recursion

```

procedure seq_blossom_recursion( $G$ ,  $M$ ,  $F$ ,  $v$ ,  $w$ )

    Form blossom:  $B = shortest\_path(F, v, w) + [v]$ 

     $G' = G$  with all blossom nodes contracted into  $w$ 

     $M' = M$  with all blossom nodes contracted into  $w$ 

     $P' = find\_aug\_path(G', M')$ 

    if  $w \in P'$  then

         $P = P'$  lifted with blossom  $B$ 

        return  $P$ 

    else

        return  $P'$ 

```

Whilst this algorithm has been valuable in solving the pairwise kidney exchange problem, the cardinality of the solutions is limited by the pairwise constraint. As research has progressed, it has become possible for the total number of kidney transplants to increase by allowing more than two patient-donor pairs to be matched as part of the optimal solution.

### 3.4.5 Implementation

In this project, Blossom's Algorithm is implemented using the Python Package Index blossomalg. This package requires a .csv file which represents the graph nodes and edges in a matrix format. The algorithm then computes the optimal solution using

Edmond's Blossom Algorithm and returns a .txt file which contains the matched edges, and the cardinality is printed in the command line. To work with this package, the datasets provided by Preflib were manipulated. The original datasets were converted from undirected graphs to directed graphs, meaning only edges where nodes were connected in each direction were preserved. The directed graph is then converted to the required matrix format and input into the package.

### **3.5 Integer Linear Programming**

Binary Integer Linear Programming (BILP) is a style of Linear Programming such that each of the decision variables may only take the values 0 or 1. Linear Programming is a valuable way to model real-world situations using linear relationships between variables to represent constraints within a model and maximising or minimising a linear function. They are usually constructed by an objective function to be maximised or minimised, as well as decision variables and linear constraints. In the case of the kidney exchange, the components are as follows:

Objective function:

Maximise the number of patients allocated a transplant in the matching

Constraints:

- Each donor can only donate one of their kidneys.
- The cycle length  $k$  must be between 2 and 3 such that  $2 \leq k \leq 3$ . (This constraint is handled before the BILP formulation as cycles greater than length 3 are filtered out before input into the formulation.)

Decision Variables:

A set of all the cycles between length 2 and 3 that are present within the graph. The BILP will decide whether the cycle is included in the solution, and therefore assigned a value of 1 or excluded from the solution and assigned a value of 0.

#### **3.5.1 Kosaraju's Algorithm**

For a directed graph  $D$ , there exists one or more strongly connected components. A strongly connected component (SCC) exists where for each vertex  $v$  in the component  $C$ , it is possible to access every other vertex  $v_x$  in  $C$ . Within a directed graph  $D$ , cycles are contained within SCCs. For a graph representing a kidney exchange, SCCs are instrumental to identifying a solution in optimal time as it means

that edges which are not part of a strongly connected component can be ignored. Specifically, in the case of the UK kidney exchange, where there may be up to 5,000 vertices, identifying transplants which cannot form part of a cycle is important to save time and resources. Importantly, SCCs do not identify an optimal solution, but exclude edges which cannot be part of the solution. Kosaraju's Algorithm has a linear time complexity, running in  $O(V+E)$  where  $V$  is the number of nodes and  $E$  is the number of edges in the graph [4].

For this project, Kosaraju's Algorithm is used to identify strongly connected components as the algorithm runs in linear time and is based upon depth first search (DFS). The algorithm is based on the idea that, if a vertex  $v$  is accessible from a vertex  $u$ , then vertex  $u$  must also be accessible via vertex  $v$ . For vertices where this is the case, they are defined as strongly connected and form part of a component.

### 3.5.2 Kosaraju's Algorithm Steps

1. Complete DFS graph traversal, pushing the source vertex onto the stack when recursive traversal for adjacent vertices is complete.
2. Reverse the direction of each edge in the graph to compute the transposed graph.
3. Complete DFS on the transposed graph with each of the vertices from the stack until the stack is empty. The nodes visited within the DFS form a strongly connected component. Where there are nodes that remain unvisited, there are more strongly connected components within the graph. Repeat this step until all nodes are visited.

The following pseudocode is inspired by OpenGenus IQ [9].

### 3.5.3 Kosaraju's Algorithm Pseudocode

```
Set  $s$  to None  
  
order = []  
  
def dfs_loop( $G$ ):  
    for node in  $G$   
        if node not explored then  
             $s$  = node  
            dfs( $G$ , node)
```

```

def dfs(G, v):

    set v.explored to True

    set v.leader to s

    for edge (v, w)

        if w not explored then

            dfs(G, w)

    set order to [v] + order

def kosaraju(G):

    dfs_loop(G_reversed)

    dfs_loop(G) in order

```

### 3.5.4 Example

Consider the directed graph  $D$  with unweighted edges  $e_1$  to  $e_{11}$  as shown below in Figure 8. As per Kosaraju's Algorithm, there are 3 strongly connected components within this graph. The resulting strongly connected components are as follows: 6, (0, 1, 4, 5, 3), 2 and are demonstrated by Figure 8 below. In this example, there are two components which only contain one vertex. This means that patient-donor pairs  $p_2d_2$  and  $p_6d_6$  cannot be matched as part of an optimal matching cycle, and therefore edges  $e_4$ ,  $e_6$  and  $e_7$  can be ignored.

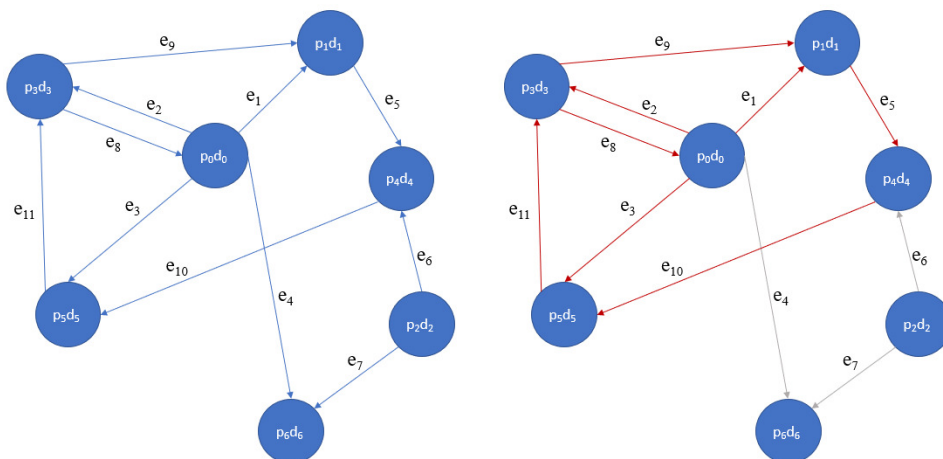


Figure 8: Strongly Connected Components

For this project, Kosaraju's algorithm has been adapted to accommodate the selected datasets. The original implementation, written in Python, was provided by Programiz.com [21].

Originally, this program created a graph from a large number of function calls to add the vertices and edges. This implementation has been adapted so that the patient-donor vertices from the dataset are created by a for-loop. Consequently, this implementation is suitable for all of the chosen datasets. Additionally, this program printed the resulting strongly connected components without storing them. This has been adapted such that the strongly connected components are stored to a 2-dimensional list which is returned from the method `print_scc`. This allows retrieval of the strongly connected components from another class in a suitable data structure to use in the implementation of Johnson's Algorithm that follows.

### 3.5.5 Johnson's Algorithm

One of the required inputs for the linear integer programming formulation of the kidney application is all of the 2- and 3-cycles that exist within the graph. The first step in computing this is to identify the strongly connected components such that all existing cycles in the graph can be located. For this implementation of the kidney exchange problem, the cycle length  $k$  is limited to a maximum of 3 such that  $2 \leq k \leq 3$ , as per the constraints required by the NHS. At this stage, a valid cycle must contain at least two nodes as the patient and donor within a patient-donor pair are not a match to each other. This simulates the current exchange as deployed by the NHS. The impact of introducing compatible patient-donor pairs into the matching pool is explored in Chapter four. Initially, all the cycles within the strongly connected components are identified using Johnson's Algorithm, then any cycles where  $k > 3$  are removed.

Johnson's Algorithm identifies all elementary cycles within a directed graph. It operates by taking each strongly connected component within a graph and identifying paths between the vertices which start and end at the same vertex. This method relies on the fact that cycles can only exist within strongly connected components, thereby allowing some edges to be ignored. Johnson's Algorithm achieves a time complexity of  $O((V + E)(|Cycle_v| + 1)) \sim O(V\Delta|Cycle_v|)$  where  $|Cycle_v|$  is the number of cycles in the graph,  $V$  is the number of vertices on the graph and  $E$  is the number of edges. In the worst case,  $|Cycle_v| = O(V!)$  which can impact performance when the dataset is large [5].

### 3.5.6 Johnson's Algorithm Steps

1. Starting at the least vertex, add this vertex to a stack  $s$  and a blocked set  $b$ .
2. Explore the first neighbour of the current vertex, adding this vertex to the stack  $s$  and blocked set. Repeat until a node is exhausted of neighbours.
3. When a vertex has no more neighbours and does not form part of a cycle, it must be blocked. Remove this vertex from the stack and blocked set, add to a blocked map  $m$  such that if a neighbour of the vertex is unblocked, this node must be too.
4. Repeat until each vertex has been explored as the starting vertex.

### 3.5.7 Johnson's Algorithm Pseudocode

**begin**

**integer list array**  $A_k(n)$ ,  $B(n)$ ; **logical array** blocked ( $n$ ); **integer**  $s$ ;

**logical procedure** CIRCUIT (**integer value**  $v$ );

**begin logical**  $f$ ;

**procedure** UNBLOCK (**integer value**  $u$ );

**begin**

blocked ( $u$ ) := false;

**for**  $w \in B(u)$  **do**

**begin**

delete  $w$  from  $B(u)$ ;

**if** blocked( $w$ ) **then**  
UNBLOCK( $w$ );

**end**

**end** UNBLOCK

$f$  := false;

stack  $v$ ;

blocked( $v$ ) := true;

L1: **for**  $w \in A_k(v)$  **do**

**if**  $w=s$  **then**

**begin**



```

        output circuit composed of
        stack followed by  $s$ ;

         $f := \text{true}$ ;

    end

    else if  $\neg \text{blocked}(w)$  then

        if  $\text{CIRCUIT}(w)$  then  $f := \text{true}$ ;

L2:    if  $f$  then  $\text{UNBLOCK}(v)$ 

        else for  $w \in A_K(v)$  do

            if  $v \notin B(w)$  then put  $v$  on  $B(w)$ ;

            unstack  $v$ ;

             $\text{CIRCUIT} := f$ ;

        end CIRCUIT;

    empty stack;

     $s := 1$ ;

    while  $s < n$  do

        begin

             $A_k :=$  adjacency structure of strong component  $K$  with
            least vertex in subgraph of  $G$  induced by  $\{s, s + 1, \dots, n\}$ ;

            if  $A_k \neq \emptyset$  then

                begin

                     $s :=$  least vertex in  $V_k$ ;

                    for  $i \in V_k$  do

                        begin

                             $\text{blocked}(i) := \text{false}$ ;

                             $B(i) := \emptyset$ ;

                        end;

L3:                    dummy :=  $\text{CIRCUIT}(s)$ ;

                         $s := s + 1$ ;

                    end

                end

            end

        end

```

```
else s := n;
```

```
end
```

```
end;
```

### 3.5.8 Example

For example, take the SCC identified in the example above and demonstrated by Figure 9 below.

Stack = {}, Blocked Set = {}, Blocked Map = {}

Taking  $p_0d_0$  as the lowest vertex, explore the neighbour  $p_3d_3$ .

Stack = {0, 1}, Blocked Set = {0, 1}, Blocked Map = {}

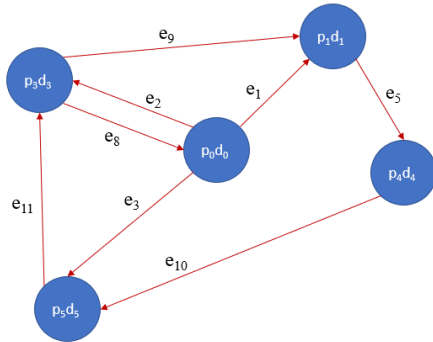


Figure 8: Strongly Connected Component Example

As 1 is not the same as the starting vertex, add it to the stack and the blocked set, and explore its neighbour 4.

Stack = {0, 1, 4}, Blocked Set = {0, 1, 4}

Repeat this following the path of each vertex first neighbour until you reach  $v_3$ .

Stack = {0, 1, 4, 5, 3}, Blocked Set = {0, 1, 4, 5, 3}

The first neighbour of  $v_3$  is  $v_0$  which matches the start vertex, and therefore a cycle has been found. Now backtrack to the next neighbour of  $v_3$ , which is  $v_1$ . As  $v_1$  is already part of the blocked set, it cannot be explored. As each of the vertices only have an out degree of 1, backtrack to the start vertex,  $v_0$ .

Repeat the process above, exploring neighbours of each vertex, starting with  $v_5$  as the neighbour of  $v_0$  and storing any completed cycles found upon each iteration.

### 3.5.9 Implementation

The implementation featured within this project uses Python code to identify all the cycles that exist within the directed graph, and then filters out cycles with length greater than three such that only the 2- and 3-cycles that are relevant to the kidney exchange implementation are returned.

### 3.5.10 ILP Formulation

One method used to model the kidney exchange is by formulating it as a binary integer linear programming problem. To achieve this, a formulation which builds upon cycle formulation, first described by Roth, Sonmez and Unver (2007) is used. [15]. The formulation is as follows:

Let  $C$  be the set of cycles that exist within a directed graph  $D$  where the cardinality of  $c_n$  is greater than or equal to 2 but does not exceed 3.

$$C = \{c_1, c_2, \dots, c_m\} \quad 2 \leq n(c_m) \leq 3.$$

Let  $b$  be a  $n \times 1$  vector which represents the upper bound of each value. In this case 1, as each donor may only donate one kidney.

$$b = [b_1, b_2, \dots, b_n] \text{ where } n = |p| \in C \text{ and } b_i = 1$$

Let  $A$  be an  $n \times m$  matrix where  $n$  equals the number of patients  $p$  included in the set of cycles  $C$ , and  $m$  is the cardinality of  $C$ .

$$A = \begin{matrix} & a_{11} & a_{12} & \dots & a_{1n} \\ & a_{21} & a_{22} & \dots & a_{2n} \\ & \vdots & \vdots & \vdots & \vdots \\ & a_{m1} & a_{m2} & \dots & a_{mn} \end{matrix} \quad n = |p| \in C \quad m = n(C)$$

Let  $A_{ij} = 1$  where  $p_i$  is in the cycle  $c_j$ .

$A_{ij} = 1$  where  $p_i \in c_j$ .

Let  $x$  be an  $n \times 1$  vector of binary variables where  $x_i = 1$  if and only if  $c_i$  belongs to the set of cycles in optimal solution  $S$ .

$$x = [x_1, x_2, \dots, x_n], x_n \in \{0, 1\} \text{ where } x_i = 1 \text{ iff } c_i \in S$$

Let  $k$  be an  $m \times 1$  vector to represent the cost of each cycle  $c_n$  within  $C$ , where  $m$  is the cardinality of  $C$ . Let  $k_n$  be the cardinality of  $c_k$ .

$$k = [k_1, k_2, \dots, k_m] \text{ where } m = n(C) \text{ and } k_m = n(c_m)$$

Now,

Maximise  $kx$  such that  $Ax \leq b$ , subject to  $x \in \{0, 1\}$

### 3.5.11 Example

For the directed graph D, there exists 10 cycles with which 2 or 3 patients belong. There are 10 unique patients belonging to the cycles.

$$C = \{(1, 3, 2, 1), (1, 3, 14, 1), (1, 13, 15, 1), (2, 11, 3, 2), (2, 12, 3, 2), (3, 14, 3), (3, 14, 13, 3), (3, 14, 15, 3), (4, 9, 4), (13, 15, 13)\}$$

Let  $b$  represent the upper bound for each donor.

$$b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

Let  $x$  be a binary  $10 \times 1$  vector such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix}$$

Let  $A$  be a  $10 \times 10$  matrix to represent each of the cycles in  $C$ .

	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$c_7$	$c_8$	$c_9$	$c_{10}$
$p_1d_1$	1	1	1	0	0	0	0	0	0	0
$p_2d_2$	1	0	0	1	1	0	0	0	0	0
$p_3d_3$	1	1	0	1	1	1	1	1	0	0
$p_4d_4$	0	0	0	0	0	0	0	0	1	0
$p_9d_9$	0	0	0	0	0	0	0	0	1	0
$p_{11}d_{11}$	0	0	0	1	0	0	0	0	0	0
$p_{12}d_{12}$	0	0	0	0	1	0	0	0	0	0
$p_{13}d_{13}$	0	0	1	0	0	0	1	0	0	1
$p_{14}d_{14}$	0	1	0	0	0	1	1	1	0	0
$p_{15}d_{15}$	0	0	1	0	0	0	0	1	0	1

Let a 10x1 vector  $k$  represent the length of each of the identified cycles.

$$k = [3 \ 3 \ 3 \ 3 \ 3 \ 2 \ 3 \ 3 \ 2 \ 2]$$

Objective function:

Maximise  $kx$

Decision variables:

$x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9$  and  $x_{10}$

Constraints:

- $Ax \leq b$
- $x_i \in \{0, 1\}$

The above formulation is implemented using the Gurobi library and used the `.optimize()` function to compute an optimal solution.

The optimal solution identified in this case selects cycles  $c_2, c_5$  and  $c_9$ . Therefore,

$$S = \{(1, 13, 15, 1), (2, 12, 3, 2), (4, 9, 4)\}.$$

In this solution,  $kx = 8$ , meaning 8 patients have been successfully matched to donors.

### 3.5.12 Limitations

As part of this project, large datasets are used to compute optimal solutions. Due to the ILP formulation selected, every cycle that exists within the graph is identified by Johnson's Algorithm, before cycles which violate cycle length constraints are filtered out. The final dataset used in this project contains 167 edges, which contains more than 70,000 cycles. Unfortunately, memory and computational power is

limited within the project and it is infeasible to compute every cycle which exists within large datasets. Therefore, this impacts the achievable results on large datasets. To overcome this problem, filtering out large cycles is carried out inside Johnson's Algorithm such that the number of identified cycles is smaller. However, rejecting large cycles earlier in the method prevents some of the cycles from ever being identified. As a result, in the case of large datasets, only an approximation to an optimal solution can be identified. The extent to which the approximation provides an acceptable solution is discussed as part of the results and analysis offered in Chapter four.

### **3.6 Top Trading Cycle**

The Top Trading Cycle (TTC) is an algorithm developed for the exchange of items without exchange of money. It is attributed to David Gale and was first published by Scarf and Shapley (1974) [17]. The algorithm contains a pool of agents who are looking to exchange an item. Each agent then chooses other items from the pool which are preferable to the item they already own and lists these from most to least preferred. Initially, the algorithm starts at agent 0 and assigns them to their most favoured item. Following this, the owner of the item that agent 0 chose is assigned to their preferred item. This process continues until a cycle is formed.

#### **3.6.1 Top Trading Cycle Description**

For this project, TTC is applied to the kidney exchange to coordinate potential kidney transplants from donor to patient. For each patient-donor pair in the pool, each patient has a list of the donors for which they are a match, such that they would prefer to receive a kidney from that donor rather than remain with their own kidney. Then, based on the matching scores between each patient and potential donor, the donors are listed from the most preferable, highest matching score to the least preferable, lowest matching score. Beginning with patient  $p_0$ , each of the patients are matched to their most preferable donor until a cycle is formed.

After each potential matching assignment, the donor who has been matched must be removed from the preference list of the remaining patient's in the pool, as each donor can only donate one kidney. In addition, the patient who has selected a donor has their preferences removed from the list, as they have been assigned their most suitable donor. There are several possible consequences after this process. It is possible that a donor is no longer desirable to any remaining patients, such that they

do not exist in any of the remaining patient's preferences. In this exchange, patients only receive a kidney on the condition that their donor donates a kidney. If a donor is no longer desirable it is not possible for their associated patient to receive a kidney, and the patient-donor pair is removed from the pool. It is also possible that a patient's preference list may now be empty, such that all the donors in their preference list have either been matched or removed from the pool. This means that this patient cannot receive a kidney in this matching and subsequently the patient and their associated donor are removed from the pool.

In some cases, a match is reached and a cycle is formed. In this situation, all the patients and their corresponding donors are removed from the pool, and the process continues with the remaining patient-donor pairs to identify any further cycles in the dataset. This process continues until either all the patient donor pairs are matched or there are no further matchings between the remaining patient-donor pairs. The result is a set of cycles containing donor-patient pairs who may exchange kidneys, which is now a possible solution.

When a potential solution is found, the preference lists are reset and the algorithm runs again now starting one patient along from the last run. For example, on the second run of the algorithm, patient  $p_1$  is matched to their preferred kidney first. This results in a different set of exchange cycles, which may or may not have a higher cardinality or total matching score than the previous identified solution. When every potential solution has been identified, an optimal solution is selected based on maximising cardinality and matching score total.

It is possible for a chain of patient-donor assignments to fail to result in a closed cycle. This may occur when the patient who is next to select their kidney no longer has any preferred kidneys left in the pool. In the event of this, all patient-donor pairs are returned to the pool and the algorithm continues one patient further along than it started the last time.

### **3.6.2 Pseudocode**

Below is the pseudocode for TTC with application to the kidney exchange.

```
set current patient to patient 0

while there are patients to explore

    if patient has an empty donor preference list then

        if current patient's donor has not been matched then
```

```

no cycle has been found

reset the donor preferences to the original list

reset matches to beginning of current cycle

iterate the patient number

continue

else

store current matching

store current state of donor preferences

continue with next patient

assign current patient to its first preference donor

clear current patient's preferences

remove matched donor from other patient's list of preferred
donors

remove any patients with an empty donor preference list

remove deleted patients from other's preference list

set current patient to patient associated with matched donor

```

### 3.6.3 Example

Given a set of patients  $S$ , such that  $S = \{s_0, s_1, s_2, s_3, s_4, s_5\}$ , and a set of donors  $D$ , such that  $D = \{d_0, d_1, d_2, d_3, d_4, d_5\}$ , for each patient  $s_x$ , there exists a subset  $s_{px}$  of  $D$  such that each donor in  $s_{px}$  is a match for patient  $s_x$ . The donors in  $s_{px}$  are ordered by the preference of  $d_x$  to  $s_x$ . The resulting graph is shown in Figure 10 below.

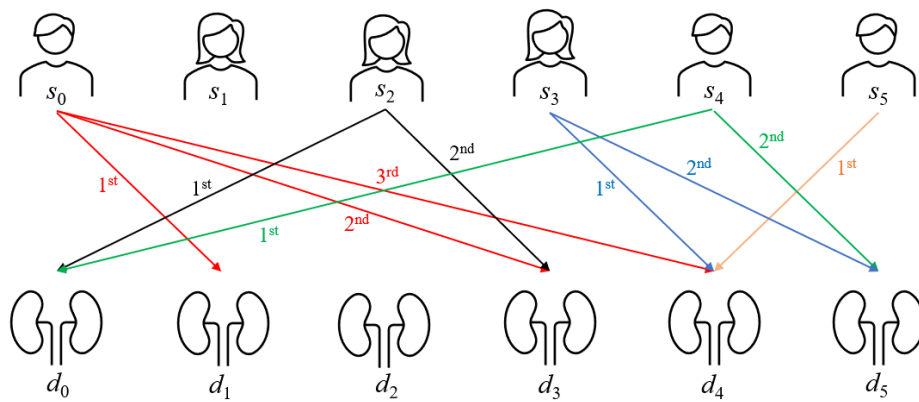


Figure 9: Top Trading Cycle Example 1



Patient  $s_1$  is not a match for any of the donors in the pool, therefore patient  $s_1$  and its corresponding donor  $d_1$  are removed from the pool as they cannot be allocated as part of an optimal matching. Similarly, donor  $d_2$  is not a match for any of the patients in  $S$ , therefore donor  $d_2$  and the corresponding patient  $s_2$  are also removed from the pool.

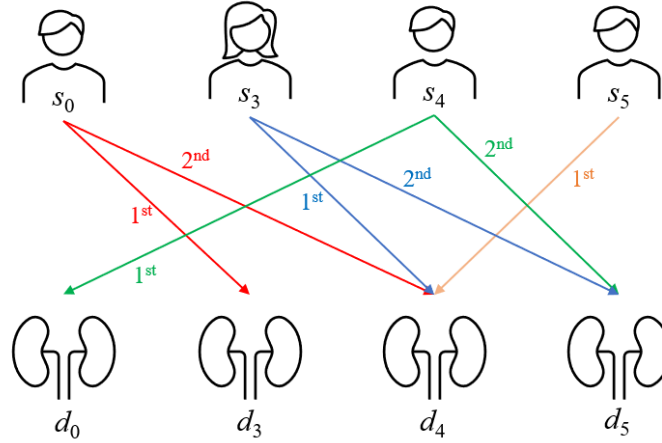


Figure 10: Top Trading Cycle Example 2

Please note, if any of the remaining patient-donor pairs were left without any preferred donors, or other patients did not prefer their donor, they would also be removed.

Now the algorithm begins assigning the patients to their preferred donor, starting with patient  $s_0$ .

The result of the allocations is as follows:

$$a_1 = s_0 \rightarrow d_3, s_3 \rightarrow d_4, s_4 \rightarrow d_0$$

The resulting allocations form a cycle, such that the patient and donor of pairs  $p_0d_0$ ,  $p_3d_3$  and  $p_4d_4$  are all featured in the matching.

This step is repeated until each of the donors have been allocated as the first patient in the matching. The resulting allocations are shown below.

$$a_2 = s_3 \rightarrow d_4, s_4 \rightarrow d_0, d_0 \rightarrow d_3$$

$$a_3 = s_4 \rightarrow d_0, s_0 \rightarrow d_3, s_3 \rightarrow d_4$$

$$a_4 = s_5 \rightarrow d_4, s_4 \rightarrow d_0, s_0 \rightarrow d_3, s_3 \rightarrow d_5$$

Note: in the above example, each of the allocations in A form a cycle and are possible solutions, where this is not the case, allocations that do not form a cycle are rejected.

The allocations  $a_1$ ,  $a_2$  and  $a_3$  all feature the same cycle of patient-donor pairs  $s_0d_0$ ,  $s_3d_3$  and  $s_4d_4$  where each patient is allocated to its first choice donor. In these matchings, patient  $s_5$  is not allocated a kidney.

Allocation  $a_4$  provides the optimal solution with a cardinality of 4, including all the patient-donor pairs in the pool. Patients  $s_5$ ,  $s_4$  and  $s_0$  are matched to their most compatible kidney, and patient  $s_3$  is matched to its second choice kidney.

### 3.6.4 Extending the Dataset

The matching score between a patient and donor is made up from computations of the following categories: HLA mismatch, matching run points the patient has taken part in, level of sensitisation and age difference between the donors. The Preflib dataset provided values for the level of sensitisation for each patient, however the other data was not available. As a result, the dataset was extended such that the matching score could be computed accurately. The previous matching run points were randomly generated values between 0 and 14, with a mean of 10. This models patients with waiting list times from less than 3 months up to 3 years and 6 months, with an average waiting time of 3 years such that the data reflects the current reality. The HLA mismatch values were generated randomly for each matching patient-donor from the set  $\{0, 5, 10, 15\}$  as these are the values which represent HLA mismatch. The ages of both the patient and the donor were generated randomly ranging from 18 to 70. The minimum value of 18 reflects the scope of the project focussing on the adult transplant waiting list. Although there is no age limit for transplant approval, it is less likely for patients over 70 to obtain approval due to the increased risks posed by surgery, which is reflected by the upper limit of 70 within the dataset. From the extended dataset, the matching score for each matching patient-donor pair is computed from the following formula:

$$\text{Score} = 50P + \frac{S}{2} + H + A$$

where  $P$  = previous matching run points,  $S$  = sensitisation levels,  $H$  = HLA mismatch and  $A = 0$  where patient-donor age difference is less than 20,  $A = 3$  otherwise. This mirrors the matching score currently implemented by the NHS.

The calculated matching scores are then used as the weight for the edge between the relevant patient-donor matching.

In addition, the dataset has been extended to represent the stage of kidney disease, and therefore indicate the level of illness in each patient. Under the NHS, patients are advised to consider transplant as a treatment option when their CKD reaches stage 4. Therefore, each patient has been assigned a stage of either 4 or 5 (end stage).

### **3.6.5 Implementation**

In this project, the implementation of this algorithm is extended to trial different methods of prioritising patients and selection of kidneys. Whilst it is true that the patient-donor matches with the highest matching score have the highest chance of a successful transplant, it is possible for each patient to be assigned to any donor that they are a match for. Based on this principle, this project explores the results when other factors were prioritised, such as the most unwell patients and patients who were more difficult to match.

One extension made to the TTC implementation is to extend the dataset to categorise each of the patients into degrees of illness caused by their kidney disease. To model this principle within the algorithm, each cycle is assigned a score based on how many stage 5 patients are allocated a kidney within that solution. According to this, the algorithm prioritises solutions where end stage patients are allocated a kidney over stage 4 patients.

Another extension made to the TTC algorithm is to prioritise patients who are difficult to match. For this implementation, difficulty to match is defined by the sensitisation level of the patient from the dataset provided. This value represents the percentage of the population that the donor is likely to have antibodies against. Therefore, to prioritise these patients two alterations have been made to the TTC algorithm. The prioritisation of highly sensitised patients is implemented by assigning each solution a value calculated from the sum of the sensitisation levels of the patients in the matching. Then, the algorithm selects the optimal solution based on the matchings with the highest levels of sensitised patients. One potential side effect of this method is that matchings with highly sensitised patients are more likely to fall through post matching but before surgery. This is due to complications in the final testing process between the patient and donor before the transplant is carried out.

As per the regulations adopted by the NHS, the maximum cycle length for transplants within the United Kingdom is 3. However, globally it is feasible for larger cycle lengths to occur. For example, the longest cycle completed in the Netherlands involved 6 pairs and 12 operations. To account for this within the project, a user-input field is included to define the length of the cycle allowed. This value,  $k$ , is used as an upper limit to the number of patients included in a cycle that forms part of a potential solution. In the code written for this project, when a cycle is identified, the length of the cycle is checked to ensure that it does not exceed the amount allowed by  $k$ . In the case where the cycle length is greater than  $k$ , the solution is rejected. Alternatively, it is accepted as part of a potential solution.

### 3.7 Top Trading Cycle and Chains

Non-directed altruistic donors are donors who do not name or specify the intended recipient of their kidney. This means that an additional kidney is available within the pool, without a paired patient. NDADs create domino chains which facilitate additional transplants that would not otherwise be possible. Within these chains, the donor who is paired with the patient receiving the NDAD kidney, donates their kidney to the kidney transplant waiting list.

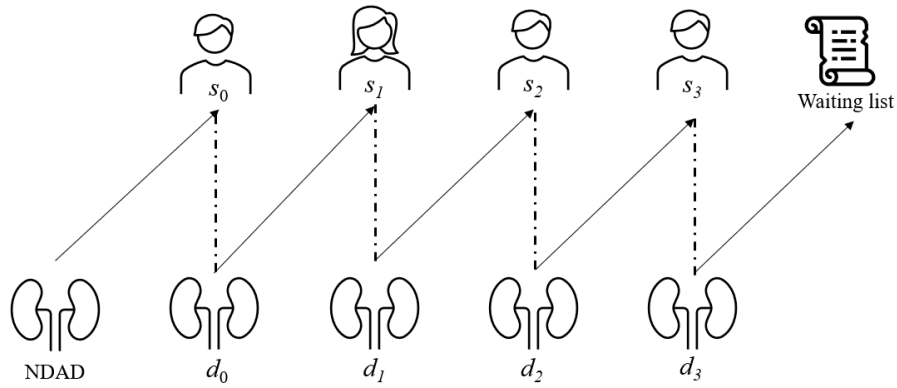


Figure 11: Altruistic Donor Chain

As demonstrated by Figure 12, four additional transplants could be allocated due to the NDAD's donation. This is because in the case where there are no existing cycles between patients  $s_0$ ,  $s_1$ ,  $s_2$  and  $s_3$  no transplants could be allocated if the altruistic donor had donated straight to the waiting list.

To model this within my dataset and implementation, each of the altruistic donors in the matching pool are assigned a dummy patient, who is marked as compatible with every donor in the pool. This is because the remaining kidney will be donated to the

waiting list, where a suitable match will be available regardless of existing factors such as blood type. Therefore, when a cycle is formed including a NDAD, it is actually a domino chain.

Another benefit to creating domino chains is that, unlike cycles, transplants allocated as part of a chain do not need to be performed simultaneously. This is due to the lack of risk of donor withdrawal, as NDADs are not expecting anything in return for the donation of their kidney. This means that the length of the chain is not restricted as with a cycle.

### **3.7.1 Example**

Let  $\{d_0: s_1, d_1: s_2, d_2: s_3, d_3: s_0\}$  be an optimal solution computed by TTCC, where  $s_0d_0$ ,  $s_1d_1$  and  $s_3d_3$  are patient donor pairs, and  $d_2$  is a NDAD. This solution represents the following action:

- Donor  $d_0$  donates their kidney to patient  $s_1$
- Donor  $d_1$  donates their kidney to patient  $s_2$ , which represents a donation to the kidney transplant waiting list
- Altruistic donor  $d_2$  donates their kidney to patient  $s_3$
- Donor  $d_3$  donates their kidney to patient  $s_0$

## 4 Results and Evaluation

### 4.1.1 Dataset 1

The first dataset that will be used to produce a matching for analysis is a reduction of the 25<sup>th</sup> dataset as provided in the sets of data by Preflib, meaning that the altruistic donors have been removed. This dataset is small, containing 16 patients, and no altruistic donors. There are 56 edges connecting the patients and donors in the graph.

Table 1 shows a sample of the graph data for graph D<sub>1</sub> which belongs to this dataset.

Patient	Donor	HLA
0	15	10
0	5	15
0	3	0
0	13	10
0	12	5
0	7	5
0	4	0
0	14	15
0	11	5
0	6	10
1	9	5

Table 1: Sample of Graph Data for Dataset D<sub>1</sub>

	Patient-Blood	Donor-Blood	%Pra	Out-Deg	Altruist	Previous Matching Run Points	Patient-Age	Donor-Age
0	O	O	0.45	10	0	7	69	23
1	A	O	0.9	12	0	9	67	25
2	O	B	0.45	0	0	9	60	37
3	O	B	0.05	1	0	11	52	62

4	O	B	0.05	1	0	11	63	59
5	O	A	0.05	4	0	10	30	44
6	B	O	0.2875	11	0	9	27	35
7	O	A	0.05	3	0	11	39	23
8	O	A	0.45	3	0	9	57	31
9	O	A	0.9	4	0	10	46	66
10	A	A	0.45	2	0	11	22	33
11	A	AB	0.05	0	0	11	53	52
12	O	A	0.05	2	0	8	24	68
13	A	B	0.2875	1	0	11	66	47
14	O	A	0.45	2	0	11	50	35
15	A	B	0.05	0	0	8	66	24

Table 2: Patient-Donor Details for Pairs in Dataset D<sub>1</sub>

#### 4.1.2 Dataset 2

The second dataset that will be used to produce a matching for analysis is the 25<sup>th</sup> dataset as provided by Preflib. The dataset includes 16 patients as in dataset 1, with the addition of 2 altruistic donors, 18 in total. As a result of this, there are 49 additional edges, 105 in total.

Table 3 shows a sample of the additional edges in the graph D<sub>2</sub>.

Patient	Donor	HLA
15	17	0
15	16	0
16	5	10
16	2	15
16	14	10
16	7	15
16	8	10

16	4	5
16	12	10
17	6	0
17	12	5

Table 3: Sample of Additional Edges in Dataset  $D_2$

The patients contained in the second dataset  $D_2$  are the same as in dataset  $D_1$ , with the addition of altruistic donors and their corresponding dummy patients,  $p_{16}$  and  $p_{17}$ .

	Patient-Blood	Donor-Blood	%Pra	Out-Deg	Altruist	Previous Matching Run Points	Patient-Age	Donor-Age
16	B	O	0.05	7	1	9	70	36
17	A	B	0.05	10	1	12	35	66

Table 4: Altruistic Donor Information for Dataset  $D_2$

#### 4.1.3 Dataset 3

Dataset  $D_3$  contains the data from dataset  $D_2$ , however it includes one final extension: the introduction of compatible patient-donor pairs into the matching pool. As a result, this dataset contains 21 patient-donor pairs, including 3 compatible patient-donor pairs. There are 62 additional edges on the graph, 167 in total. Table 5 shows the patient-donor information for the compatible patient-donor pairs.

	Patient-Blood	Donor-Blood	%Pra	Out-Deg	Altruist	Previous Matching Run Points	Patient-Age	Donor-Age
18	O	O	0.05	4	0	0	48	32
19	A	O	0.05	12	0	0	25	34
20	AB	B	0.05	21	0	0	46	52

Table 5: Compatible Patient-Donor Pair Information for Dataset  $D_3$

Table 6 shows a sample of the information on the matching relations between the additional and existing patients.



Patient	Donor	HLA
18	0	5
18	18	0
18	1	0
18	6	10
19	5	10
19	7	0
19	18	5
19	8	5
19	9	5
19	10	0

Table 6: Compatible Patient-Donor Matching Information for Dataset D<sub>3</sub>

#### 4.2.1 Pairwise Exchange Results Dataset 1

The directed graph which represents dataset D<sub>1</sub> is shown below in Figure 13.

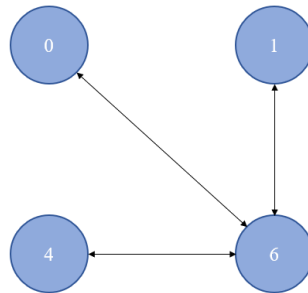


Figure 12: Directed Graph for Dataset D<sub>1</sub>

The directed graph can also be represented as a matrix:

	0	1	4	6
0	0	0	0	1
1	0	0	0	1
4	0	0	0	1
6	1	1	1	0

When this data is input into the Blossom Algorithm, the following result is obtained:

```
C:\Users\molly\OneDrive\Documents\year 3\FYP>blossalg infile.csv blossom.txt
There are 2 matched nodes in maximum matched graph.
```

Figure 13: Screenshot of Blossom Algorithm Output

The resulting output file contains:

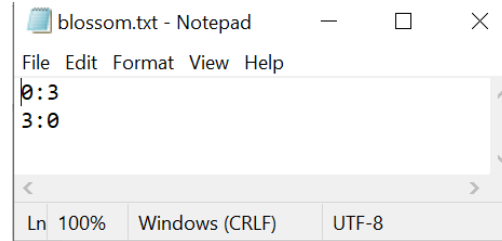


Figure 14: Screenshot of Blossom Output File

This means that the first node in the graph should be matched to the third node in the graph. Therefore, the optimal solution is as follows:

- Donor  $d_0$  donates their kidney to patient  $p_6$
- Donor  $d_6$  donates their kidney to patient  $p_0$

This solution has a cardinality of 2. As a result, 14 willing donors are unable to donate, along with 14 patients who are unable to receive as part of this matching.

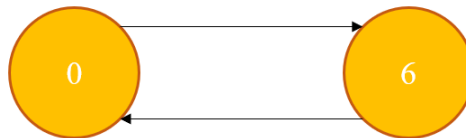


Figure 15: Pairwise Exchange Optimal Matching for Dataset  $D_1$

#### 4.2.2 Pairwise Exchange Results Dataset 2

The directed graph produced from dataset  $D_2$  is shown in Figure 17 below:

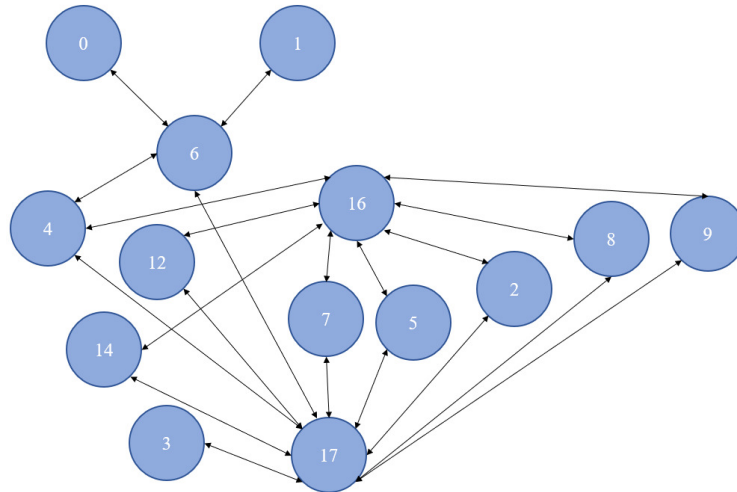


Figure 16: Directed Graph for Dataset D<sub>2</sub>

The graph, shown in Figure 17, can be represented as a matrix, shown below:

	0	1	2	3	4	5	6	7	8	9	12	14	16	17
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	1	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	1	0	0	0	0	0	1	1
5	0	0	0	0	0	0	0	0	0	0	0	0	1	1
6	1	1	0	0	1	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	0	0	0	0	1	1
8	0	0	0	0	0	0	0	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	0	0	0	0	0	1
12	0	0	0	0	0	0	0	0	0	0	0	0	1	1
14	0	0	0	0	0	0	0	0	0	0	0	0	1	1
16	0	0	1	0	1	1	0	1	1	0	1	1	0	0
17	0	0	1	1	1	1	1	1	1	1	1	1	0	0

The output from the Blossom Algorithm is shown in Figures 18 and 19 below:

```
C:\Users\molly\OneDrive\Documents\year 3\FYP>blossalg infile.csv blossom.txt
There are 6 matched nodes in maximum matched graph.
```

Figure 17: Screenshot of Output from Blossom Algorithm for Dataset D<sub>2</sub>

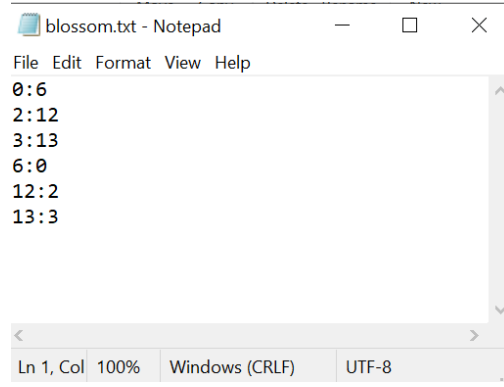


Figure 18: Screenshot of Blossom Algorithm Output File

This output shows that the optimal solution is as follows:

- Donor  $d_0$  donates their kidney to patient  $p_6$
- Donor  $d_2$  donates their kidney to the active waiting list
- Donor  $d_3$  donates their kidney to the active waiting list
- Donor  $d_6$  donates their kidney to patient  $p_0$
- Donor  $d_{16}$  donates their kidney to patient  $p_2$
- Donor  $d_{17}$  donates their kidney to patient  $p_3$

As a result, the cardinality of this solution is 6, as there are 4 patients from the matching pool and two patients from the active waiting list who are allocated a kidney.

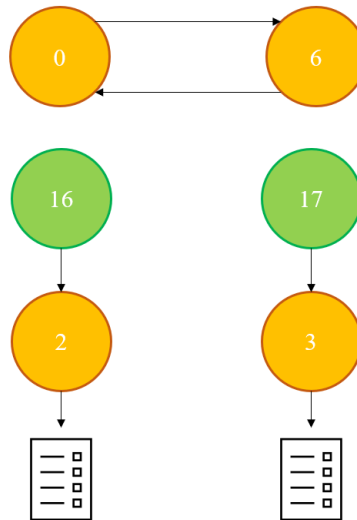


Figure 19: Pairwise Exchange Optimal Solution for Dataset D2

#### 4.2.3 Pairwise Exchange Results Dataset 3

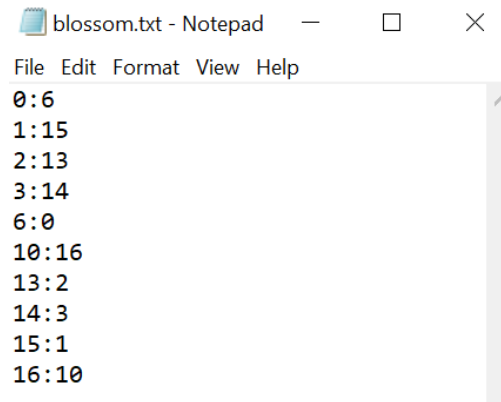
The directed graph produced from dataset  $D_3$  has 57 edges connecting 18 donor-patient nodes. Therefore, the resulting graph is large and viewing the graph does not aid understanding and is omitted as a result.

	0	1	2	3	4	5	6	7	8	9	10	12	14	16	17	18	19	20
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
6	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	1
7	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
16	0	0	1	1	1	1	0	1	1	0	0	1	1	0	0	0	0	1
17	0	0	1	0	1	1	1	1	1	1	0	1	1	0	0	0	0	0
18	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
19	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
20	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1

The output from the Blossom Algorithm is shown in Figure 21 below:

```
C:\Users\molly\OneDrive\Documents\year 3\FYP>blossalg infile.csv blossom.txt
There are 10 matched nodes in maximum matched graph.
```

Figure 20: Screenshot of Blossom Algorithm Output for Dataset D<sub>3</sub>



```
blossom.txt - Notepad
File Edit Format View Help
0:6
1:15
2:13
3:14
6:0
10:16
13:2
14:3
15:1
16:10
```

Figure 21: Screenshot of Output File for Blossom Algorithm Dataset D<sub>3</sub>

This output shows that the optimal solution is as follows:

- Donor  $d_0$  donates their kidney to patient  $p_6$
- Donor  $d_1$  donates their kidney to patient  $p_{15}$
- Donor  $d_2$  donates their kidney to patient  $p_{13}$

- Donor  $d_3$  donates their kidney to patient  $p_{14}$
- Donor  $d_6$  donates their kidney to patient  $p_0$
- Donor  $d_{10}$  donates their kidney to the active waiting list
- Donor  $d_{13}$  donates their kidney to patient  $p_2$
- Donor  $d_{14}$  donates their kidney to patient  $p_3$
- Donor  $d_{15}$  donates their kidney to patient  $p_1$
- Donor  $d_{16}$  donates their kidney to patient  $p_{10}$

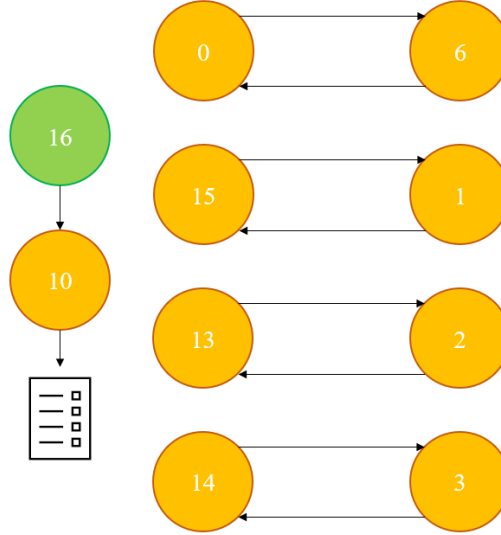


Figure 22: Pairwise Exchange Optimal Solution for Dataset  $D_3$

As a result, the cardinality of this solution is 10, as there are 9 patients from the matching pool and one patient from the active waiting list who are allocated a kidney.

#### 4.3.1 ILP Results Dataset 1

The SCCs identified by Kosaraju's Algorithm for graph  $D_1$  are as follows:

15, 11, 12, 10, 14, 2, (0, 3, 6, 1, 4, 5, 13, 7, 9, 8)

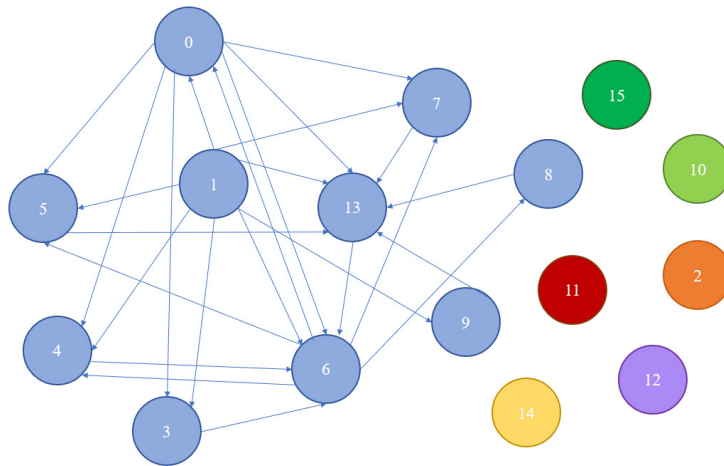


Figure 23: SCCs for Dataset  $D_1$

As a result, 30 edges were excluded from  $D_1$ , reducing the number of edges left to optimise from 58 to 28.

From the identified SCCs, Johnson's algorithm computed all the cycles within the graph where cycle length  $k$  such that  $2 \leq k \leq 3$ . There are 13 cycles, 3 of length 2, and 9 of length 3.

The set of cycles  $C$  is as follows:

$$C = \{(0, 3, 6, 0), (0, 6, 0), (0, 6, 1, 0), (0, 4, 6, 0), (0, 13, 6, 0), (1, 3, 6, 1), (1, 6, 1), (1, 4, 6, 1), (1, 13, 6, 1), (4, 6, 4), (5, 13, 6, 5), (6, 7, 13, 6), (6, 8, 13, 6)\}$$

Let  $C$  be represented by the following matrix:

	$c_0$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$c_7$	$c_8$	$c_9$	$c_{10}$	$c_{11}$	$c_{12}$
<b>0</b>	1	1	1	1	1	0	0	0	0	0	0	0	0
<b>1</b>	0	0	1	0	0	1	1	1	1	0	0	0	0
<b>3</b>	1	0	0	0	0	1	0	0	0	0	0	0	0
<b>4</b>	0	0	0	1	0	0	0	1	0	1	0	0	0
<b>5</b>	0	0	0	0	0	0	0	0	0	0	1	0	0
<b>6</b>	1	1	1	1	1	1	1	1	1	1	1	1	1
<b>7</b>	0	0	0	0	0	0	0	0	0	0	0	1	0
<b>8</b>	0	0	0	0	0	0	0	0	0	0	0	0	1
<b>9</b>	0	0	0	0	0	0	0	0	0	0	0	0	0
<b>13</b>	0	0	0	0	1	0	0	0	1	0	1	1	1

Let  $b$  represent the upper bound for each donor and  $x$  be a binary  $10 \times 1$  vector such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

$$b = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \end{bmatrix}$$

Let a 13x1 vector  $k$  represent the length of each of the identified cycles.

$$k = [3 \ 2 \ 3 \ 3 \ 3 \ 3 \ 2 \ 3 \ 3 \ 2 \ 3 \ 3 \ 3]$$

Now, maximise  $kx$ , with respect to  $Ax \leq b$  and  $x_i \in \{0, 1\}$ .

When using the Gurobi Optimiser, the following output is returned:

```
Optimize a model with 9 rows, 13 columns and 36 nonzeros
Model fingerprint: 0x133a9aee
Variable types: 0 continuous, 13 integer (13 binary)
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 3e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 3.0000000
Presolve removed 9 rows and 13 columns
Presolve time: 0.00s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.00 seconds
Thread count was 1 (of 4 available processors)

Solution count 1: 3

Optimal solution found (tolerance 1.00e-04)
Best objective 3.000000000000e+00, best bound 3.000000000000e+00, gap 0.0000%
x[0] = 1
x[1] = 0
x[2] = 0
x[3] = 0
x[4] = 0
x[5] = 0
x[6] = 0
x[7] = 0
x[8] = 0
x[9] = 0
x[10] = 0
x[11] = 0
x[12] = 0
```

Figure 24: Screenshot of ILP Results for Dataset D<sub>1</sub>

The returned cycle which forms the optimal solution is  $c_0$ . Cycle  $c_0$  has a cardinality of 3, meaning that 3 patients are allocated a kidney transplant in total as part of the optimal solution. It is of note that in this particular circumstance, there are 8 other cycles which also have a cardinality of 3, therefore, to choose the optimum cycle,



the total weight of each cycle should be considered. A limitation of this program is that cardinality is the determining factor in choosing an optimal matching, and total weight is not considered. This means that whilst  $c_0$  was selected as the optimal solution, any of the cycles length 3 provide the same cardinality.

Cycle  $c_0 = (0, 3, 6, 0)$ . This represents the following action:

- Donor  $d_0$  donates their kidney to patient  $p_3$
- Donor  $d_3$  donates their kidney to patient  $p_6$
- Donor  $d_6$  donates their kidney to patient  $p_0$

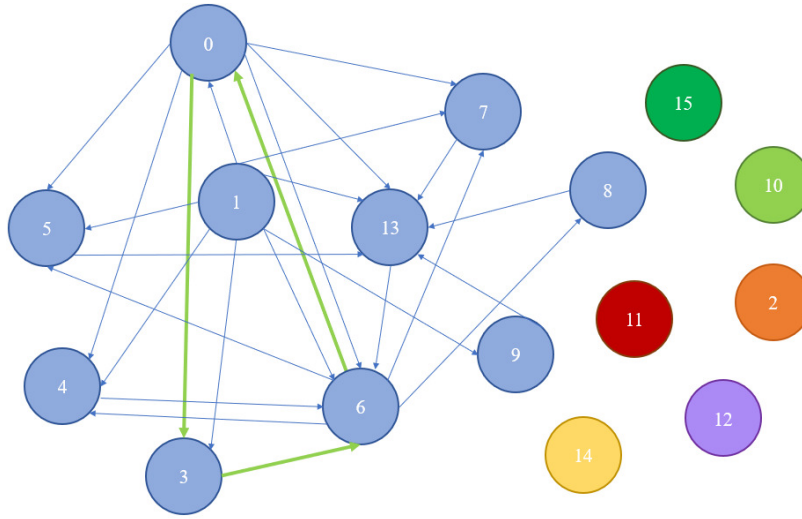


Figure 25: Optimal ILP Solution for Dataset  $D_1$

This dataset was also used with the cycle length  $k$  constraint such that  $2 \leq k \leq 3$ , but without the chain length constraint. As transplants from domino chains are not required to occur simultaneously, the need for a small chain is not always necessary.

The SSCs identified from graph  $D_1$ , are the same as in the results above. However, no additional chains were identified. Therefore, the results in this case are the same as above.

Additionally, this dataset was also used with the ILP formulation without the cycle length constraint such that cycle length =  $k$  where  $2 \leq k \leq 3$ . This set of results is to highlight the difference in cardinality of the maximum matching and state the case for research into the practicalities of cycles containing more patients.

The SCCs identified within the graph  $D_1$ , are the same as above. From the identified SCCs, Johnson's Algorithm computed all the cycles within  $D_1$ . There are now 23 cycles within the set  $C$ , shown below:

$$C = \{(0, 3, 6, 0), (0, 3, 6, 1, 0), (0, 6, 0), (0, 6, 1, 0), (0, 4, 6, 0), (0, 4, 6, 1, 0), (0, 5, 13, 6, 0), (0, 5, 13, 6, 1, 0), (0, 13, 6, 0), (0, 13, 6, 1, 0), (0, 7, 13, 6, 0), (0, 7, 13, 6, 1, 0), (1, 3, 6, 1), (1, 6, 1), (1, 4, 6, 1), (1, 5, 13, 6, 1), (1, 13, 6, 1), (1, 7, 13, 6, 1), (1, 9, 13, 6, 1), (4, 6, 4), (5, 13, 6, 5), (6, 7, 13, 6), (6, 8, 13, 6)\}$$

Now as explained in the detailed ILP formulation in Chapter 3.5, maximise  $kx$ , with respect to  $Ax \leq b$  and  $x_i \in \{0, 1\}$ .

When using the Gurobi Optimiser, the following output is returned:

```
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 5e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 3.0000000
Presolve removed 10 rows and 23 columns
Presolve time: 0.00s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.00 seconds
Thread count was 1 (of 4 available processors)

Solution count 2: 5 3

Optimal solution found (tolerance 1.00e-04)
Best objective 5.000000000000e+00, best bound 5.000000000000e+00, gap 0.0000%
x[0] = 0
x[1] = 0
x[2] = 0
x[3] = 0
x[4] = 0
x[5] = 0
x[6] = 0
x[7] = 1
x[8] = 0
x[9] = 0
x[10] = 0
x[11] = 0
x[12] = 0
x[13] = 0
x[14] = 0
x[15] = 0
x[16] = 0
x[17] = 0
x[18] = 0
x[19] = 0
x[20] = 0
x[21] = 0
x[22] = 0
```

Figure 26: Screenshot of ILP Results for Dataset D<sub>1</sub> without cycle length constraints

The optimal solution is returned as cycle  $c_7$  only with a cardinality of 5, meaning that 2 additional patients are allocated a kidney compared to the solution which includes the cycle length constraint. As above, as a single cycle of length 5 has been selected, there are 2 other cycles which would also return maximum cardinality in this case. Actions required by selection of cycle  $c_7$  detailed below:

Cycle  $c_7 = (0, 5, 13, 6, 1, 0)$

- Donor  $d_0$  donates their kidney to patient  $p_5$
- Donor  $d_5$  donates their kidney to patient  $p_{13}$
- Donor  $d_{13}$  donates their kidney to patient  $p_6$
- Patient  $d_6$  donates their kidney to patient  $p_1$
- Donor  $d_1$  donates their kidney to patient  $p_0$

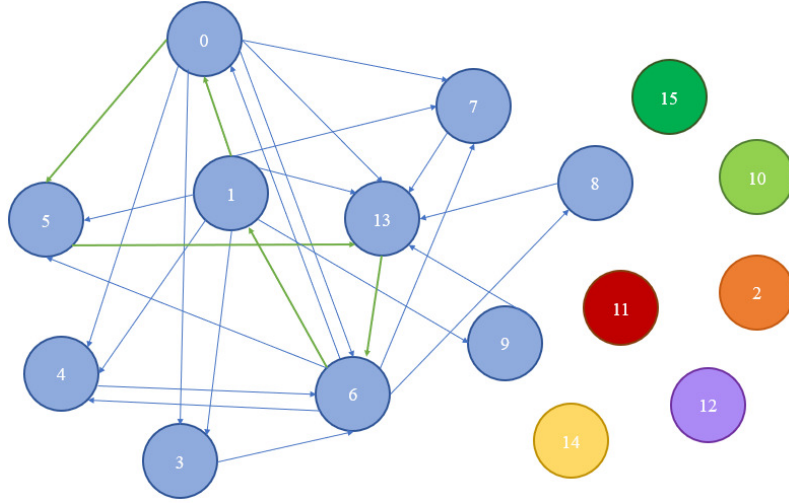


Figure 27: Optimal ILP Solution for Dataset D1 without cycle length constraints.

#### 4.3.2 ILP Results Dataset 2

The SCC identified by Kosaraju's Algorithm for graph  $D_2$  is as follows:

(0, 3, 6, 1, 2, 16, 4, 17, 5, 10, 11, 15, 13, 7, 8, 9, 12, 14)

As the altruistic donor's dummy patients are compatible with every donor, each vertex becomes accessible from every other vertex, meaning that the graph  $D_2$  is one SCC. As a result, the graph contains 18 patient-donor pairs, connected by 106 edges, for which a diagram does not aid understanding due to the vast number of edges.

From the identified SCC, Johnson's algorithm computed all the cycles within the graph where cycle length  $k$  such that  $2 \leq k \leq 3$ . There are 78 cycles, 20 of length 2, and 58 of length 3.

The set of cycles  $C$  is as follows:

$C = \{(0, 3, 6, 0), (0, 6, 0), (0, 6, 1, 0), (0, 4, 6, 0), (0, 17, 6, 0), (0, 13, 6, 0), (1, 3, 6, 1), (1, 6, 1), (1, 4, 6, 1), (1, 17, 6, 1), (1, 13, 6, 1), (2, 16, 2), (2, 17, 6, 2), (2, 17, 2), (3, 6, 17, 3), (3, 17, 3), (4, 6, 16, 4), (4, 6, 4), (4, 6, 17, 4), (4, 16, 4), (4, 17, 6, 4), (4, 17, 4), (5, 16, 5), (5, 17, 6, 5), (5, 17, 5), (5, 10, 16, 5), (5, 10, 17, 5), (5, 11, 16, 5), (5, 11, 17, 5), (5, 15, 16, 5), (5, 15, 17, 5), (5, 13, 6, 5), (5, 13, 16, 5), (5, 13, 17, 5), (6, 17, 6), (6, 10, 17, 6), (6, 11, 17, 6), (6, 15, 17, 6), (6, 7, 17, 6), (6, 7, 13, 6), (6, 8, 17, 6), (6, 8, 13, 6), (6, 12, 17, 6), (7, 16, 7), (7, 17, 7), (7, 11, 16, 7), (7, 11, 17, 7), (7, 15, 16, 7), (7, 15, 17, 7), (7, 13, 16, 7), (7, 13, 17, 7), (8, 16, 8), (8, 17, 8), (8, 10, 16, 8), (8, 10, 17, 8), (8, 11, 16, 8), (8, 11, 17, 8), (8, 13, 16, 8), (8, 13, 17, 8), (9, 17, 9), (9, 10, 17, 9), (9, 11, 17, 9), (9, 15, 17, 9), (9, 13, 17, 9), (10, 16, 14, 10), (10, 17, 14, 10), (11, 16, 12, 11), (11, 16, 14, 11), (11, 17, 12, 11), (11, 17, 14, 11), (12, 16, 12), (12, 17, 12), (12, 15, 16, 12), (12, 15, 17, 12), (14, 16, 14), (14, 17, 14), (14, 15, 16, 14), (14, 15, 17, 14)\}$

Let  $b$  represent the upper bound for each donor such that every value in  $b = 1$ , and  $x$  be a binary  $78 \times 1$  vector such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

[illegible]

When using the Gurobi Optimiser, the following output is returned:

```

Optimize a model with 18 rows, 78 columns and 214 nonzeros
Model fingerprint: 0x87ecef0e
Variable types: 0 continuous, 78 integer (78 binary)
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 3e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 7.0000000
Presolve removed 12 rows and 70 columns
Presolve time: 0.01s
Presolved: 6 rows, 8 columns, 16 nonzeros
Variable types: 0 continuous, 8 integer (8 binary)

Root relaxation: objective 9.0000000e+00, 3 iterations, 0.00 seconds

   Nodes      |   Current Node   |   Objective Bounds   |         Work
Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap   | It/Node Time
*    0       0             0       9.0000000    9.00000    0.00%   -     0s

Explored 0 nodes (3 simplex iterations) in 0.01 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 9 7

Optimal solution found (tolerance 1.00e-04)
Best objective 9.000000000000e+00, best bound 9.000000000000e+00, gap 0.0000%
x[0] = 0
x[1] = 0
x[2] = 1
x[3] = 0
x[4] = 0
x[5] = 0
x[6] = 0
x[7] = 0
x[8] = 0
x[9] = 0

```

Figure 28: Screenshot of Optimal ILP Solution for Dataset D<sub>2</sub>

The returned cycles which form the optimal solution are  $c_2$ ,  $c_{29}$  and  $c_{60}$ . The total cardinality of this solution is 9, 6 more than the cardinality of the optimal solution excluding the altruistic donors. This means that the addition of just two altruistic donors into the pool has facilitated twice as many additional allocations. The resulting action from this matching is detailed below:

Cycle  $c_2 = (0, 6, 1, 0)$ .

- Donor  $d_0$  donates their kidney to patient  $p_6$
- Donor  $d_6$  donates their kidney to patient  $p_1$
- Donor  $d_1$  donates their kidney to patient  $p_0$

Cycle  $c_{29} = (5, 15, 16, 5)$ .

- Donor  $d_5$  donates their kidney to patient  $p_{15}$
- Donor  $d_{15}$  donates their kidney to the waiting list
- Altruistic donor  $d_{16}$  donates their kidney to patient  $p_5$

Cycle  $c_{60} = (9, 10, 17, 9)$ .

- Donor  $d_9$  donates their kidney to patient  $p_{10}$
- Donor  $d_{10}$  donates their kidney to the waiting list
- Altruistic donor  $d_{17}$  donates their kidney to patient  $p_9$

As a result of this matching, 7 patients within the matching pool have been allocated a kidney, and 2 patients on the kidney transplant waiting list have also been donated a kidney.

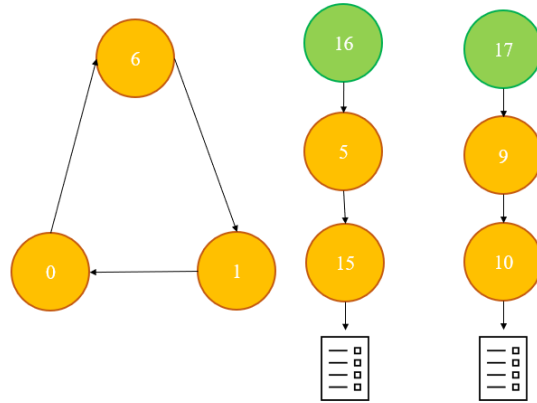


Figure 29: Optimal ILP Solution for Dataset D2, where Cycle Length Restricted

Due to the nature of altruistic domino chains, it is not required for the allocated transplants to be carried out simultaneously, as discussed in the introduction to domino chains in Chapter 3.7. Therefore, this dataset is also trialled against the constraint where cycles must be kept between length 2 and 3, but domino chains can be of any length. The SCC identified in the graph remains as above.

The set of identified cycles  $C$  where cycle length  $k$  such that  $2 \leq k \leq 3$ , and domino chains of any length contains 1075 cycles and chains. From this, the ILP formulation will be modelled as described in the examples above both earlier in this chapter, and in Chapter 3.7.

When using the Gurobi Optimiser, a sample of the output is displayed in Figure 31 below:

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 2 physical cores, 4 logical processors, using up to 4 threads
Optimize a model with 18 rows, 1075 columns and 6380 nonzeros
Model fingerprint: 0x4dbdd97b
Variable types: 0 continuous, 1075 integer (1075 binary)
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 9e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 7.0000000
Presolve removed 2 rows and 759 columns
Presolve time: 0.04s
Presolved: 16 rows, 316 columns, 1524 nonzeros
Variable types: 0 continuous, 316 integer (316 binary)

Root relaxation: objective 1.200000e+01, 61 iterations, 0.01 seconds

   Nodes      |      Current Node      |      Objective Bounds      |      Work
  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd    Gap   | It/Node Time
-----
    0     0   12.00000    0   4    7.00000    12.00000   71.4%   -    0s
H    0     0           12.0000000    12.00000   0.00%   -    0s
    0     0   12.00000    0   4   12.00000    12.00000   0.00%   -    0s

Explored 1 nodes (61 simplex iterations) in 0.09 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 12 7

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200000000000e+01, best bound 1.200000000000e+01, gap 0.0000%
x[0] = 0
x[1] = 0
x[2] = -0
x[3] = 0
x[4] = -0
:
x[1069] = -0
x[1070] = 1
x[1071] = -0
x[1072] = -0
x[1073] = 0
x[1074] = 0
```

Figure 30: Sample of Screenshot of ILP Solution for D<sub>2</sub>

The optimal solution in this case selects cycles  $c_{105}$  and  $c_{1070}$ . The cardinality of this matching is 12, meaning an additional 3 patients have been allocated a kidney than when the domino chain length was restricted.

Cycle  $c_{105} = [0, 5, 10, 11, 16, 8, 13, 6, 1, 0]$

- Donor  $d_0$  donates their kidney to patient  $p_5$
- Donor  $d_5$  donates their kidney to patient  $p_{10}$
- Donor  $d_{10}$  donates their kidney to patient  $p_{11}$
- Donor  $d_{11}$  donates their kidney to the waiting list
- Altruistic donor  $d_{16}$  donates their kidney to patient  $p_8$
- Donor  $d_8$  donates their kidney to patient  $p_{13}$
- Donor  $d_{13}$  donates their kidney to patient  $p_6$
- Donor  $d_6$  donates their kidney to patient  $p_1$
- Donor  $p_1$  donates their kidney to patient  $p_0$

Cycle  $c_{1070} = [12, 15, 17, 12]$

- Donor  $d_{12}$  donates their kidney to patient  $p_{15}$
- Donor  $d_{15}$  donates their kidney to the waiting list
- Donor  $d_{17}$  donates their kidney to patient  $p_{12}$

As a result of this matching, 10 patients from the matching pool and 2 patients from the waiting list are allocated a kidney.

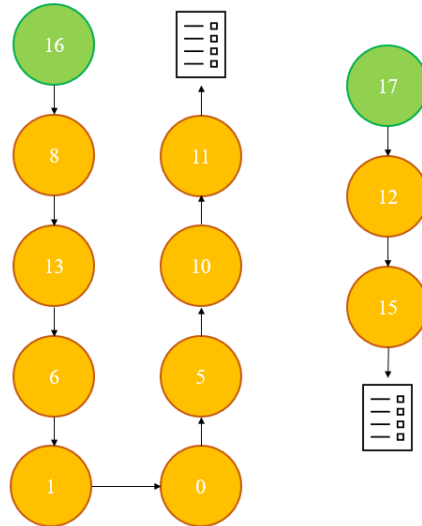


Figure 31: Optimal ILP Solution for Dataset D<sub>2</sub>, No Restrictions

Finally, ILP was used to compute the optimal solution where the cycle length  $k$ , and the domino chain length  $l$  are both unrestricted. This models the situation where particularly the constraint on cycle length is ignored, to understand the potential limitations imposed on the optimal solution further.

As a result, 10 additional cycles were identified to decipher between to compute the optimal solution.

When using the Gurobi Optimiser, a sample of the output is displayed in Figure 33 below:

```

Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 2 physical cores, 4 logical processors, using up to 4 threads
Optimize a model with 18 rows, 1085 columns and 6422 nonzeros
Model fingerprint: 0xf0a48dc1
Variable types: 0 continuous, 1085 integer (1085 binary)
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 9e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 7.0000000
Presolve removed 2 rows and 764 columns
Presolve time: 0.01s
Presolved: 16 rows, 321 columns, 1536 nonzeros
Found heuristic solution: objective 12.0000000
Variable types: 0 continuous, 321 integer (321 binary)

Root relaxation: cutoff, 79 iterations, 0.00 seconds

   Nodes |      Current Node |      Objective Bounds      |      Work
  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   cutoff     0       12.00000    12.00000   0.00%   -    0s

Explored 0 nodes (79 simplex iterations) in 0.02 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 12 7

Optimal solution found (tolerance 1.00e-04)
Best objective 1.200000000000e+01, best bound 1.200000000000e+01, gap 0.0000%
x[0] = 0
x[1] = 0
x[2] = 0
x[3] = 0
x[4] = 0
x[5] = 0
x[6] = 0
x[7] = 0
x[8] = 0
:
x[1075] = 0
x[1076] = 0
x[1077] = 0
x[1078] = 0
x[1079] = 0
x[1080] = 0
x[1081] = 0
x[1082] = 0
x[1083] = 0
x[1084] = 0

```

Figure 32: Sample of Screenshot of Optimal ILP Solution for Dataset D2, No Restrictions

The optimal matching in this case consists of cycles  $c_{105}$  and  $c_{1065}$ . The cardinality of this matching is 12, so no additional patients have been allocated a kidney. Although one of the chosen cycles differs from the previous result, this is due to an arbitrary selection of a 3-cycle matching, and therefore the difference between the two solutions is negligible. Therefore, in this case, the constraint on the cycle length  $k$ ,  $2 \leq k \leq 3$ , does not limit the optimality of the solution.



#### 4.3.3 ILP Results Dataset 3

The SCC identified by Kosaraju's Algorithm for graph  $D_3$  are as follows:

(0, 3, 6, 1, 2, 16, 4, 17, 5, 10, 11, 18, 19, 7, 13, 15, 8, 9, 12, 14, 20)

As this does not reduce the initial graph, there are still 167 edges and 21 nodes. Therefore, an image of the SCC graph does not aid understanding.

Due to the large number of matches between patients and donors, and failure to reduce the graph using Kosaraju's Algorithm, this dataset is considered large. As a result, it is not possible to store every instance of a cycle that exists within  $D_3$  due to memory and computational power resources available. Therefore, this dataset is processed according to the secondary method presented in Chapter 3.5.12.

In this instance, the initial search for cycles is limited to 10. As a result, it is possible that not every single cycle of lengths 2 and 3 were identified. Despite this, 146 cycles were identified, but due to the large number, they are not shown here.

As before, let a  $146 \times 1$  vector  $b$  represent the upper bound for each donor and let a  $146 \times 1$  vector  $x$  represent the selected cycles such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

Let an  $146 \times 1$  vector  $k$  represent the length of each of the identified cycles.

Now, maximise  $kx$ , with respect to  $Ax \leq b$  and  $x_i \in \{0, 1\}$ .

Despite limiting the cycle search, the algorithm runtime has increased from almost instantaneously to 25.4 seconds. When using the Gurobi Optimiser, the following output is returned:

```

Thread count: 2 physical cores, 4 logical processors, using up to 4 threads
Optimize a model with 21 rows, 147 columns and 420 nonzeros
Model fingerprint: 0x24b4778c
Variable types: 0 continuous, 147 integer (147 binary)
Coefficient statistics:
  Matrix range      [1e+00, 1e+00]
  Objective range   [2e+00, 3e+00]
  Bounds range      [1e+00, 1e+00]
  RHS range         [1e+00, 1e+00]
Found heuristic solution: objective 13.0000000
Presolve removed 1 rows and 21 columns
Presolve time: 0.01s
Presolved: 20 rows, 126 columns, 364 nonzeros
Variable types: 0 continuous, 126 integer (126 binary)

Root relaxation: objective 1.500000e+01, 65 iterations, 0.00 seconds

  Nodes |      Current Node |      Objective Bounds |      Work
Expl Unexpl | Obj Depth IntInf | Incumbent BestBd Gap | It/Node Time
*  0    0 |          0        | 15.000000 15.0000 0.00% | -    0s

Explored 0 nodes (65 simplex iterations) in 0.01 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 15 13

Optimal solution found (tolerance 1.00e-04)
Best objective 1.500000000000e+01, best bound 1.500000000000e+01, gap 0.0000%

x[52] = 1
x[65] = 1
x[115] = 1
x[129] = 1
x[140] = 1

```

Figure 33: Screenshot of Optimal ILP Solution for Dataset D<sub>3</sub> Restricted Cycle and Chain Length

The selected cycles are as follows:

Cycle  $c_{52} = (1, 12, 18, 1)$ , cycle  $c_{65} = (4, 6, 20, 4)$ , cycle  $c_{115} = (7, 13, 16, 7)$ , cycle  $c_{129} = (8, 11, 19, 8)$ , cycle  $c_{140} = (9, 15, 17, 9)$

This solution has a cardinality of 15, made up from 13 matching pool patient allocations and 2 active waiting list allocations. This is an improvement of 6 when compared to the ILP solution that does not include the compatible patient-donor pairs. In addition, this solution identified 3 more matches than the equivalent TTC solution. Therefore, this approach can be considered a good approximation to an optimal solution.

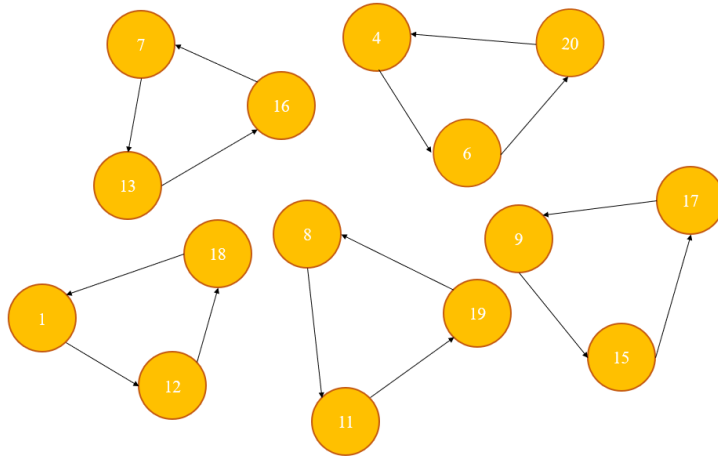


Figure 34: Optimal ILP Solution Approximation for Dataset D<sub>3</sub>

Considering only eligible cycles and chains, where cycle length  $k$  such that  $2 \leq k \leq 3$  and chain length  $l$  unrestricted and limiting the cycle length search to 10, 74,157 cycles and chains were identified. Due to the large number of cycles found, they are not listed here. Every patient-donor pair in  $D_3$  is included in one or more cycles. Therefore, let  $C$  be represented by a 74157x21 matrix.

As before, let a 21x1 vector  $b$  represent the upper bound for each donor and let a 21x1 vector  $x$  represent the selected cycles such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

Let a 74157x1 vector  $k$  represent the length of each of the identified cycles.

Now, maximise  $kx$ , with respect to  $Ax \leq b$  and  $x_i \in \{0, 1\}$ .

When using the Gurobi Optimiser, the following output is returned:

```
Presolve time: 0.46s
Presolved: 21 rows, 21631 columns, 159301 nonzeros
Variable types: 0 continuous, 21631 integer (21631 binary)

Starting sifting (using dual simplex for sub-problems)...

  Iter   Pivots   Primal Obj   Dual Obj   Time
    0         0   infinity   -2.7790000e+03   1s
    1        21   2.6859973e+09   -9.3178569e+01   1s
    2        58   1.0969989e+09   -9.2972394e+01   1s
    3        95   7.9499919e+08   -9.2978595e+01   1s
    4       133   4.4499954e+08   -9.3036881e+01   1s
    5       186  -9.0045331e+00   -9.2989138e+01   1s
    6       234  -1.2003681e+01   -2.1011144e+01   1s
    7       255  -1.4003558e+01   -2.1002527e+01   1s
    8       293  -1.7002851e+01   -1.7707688e+01   1s

Sifting complete

Root relaxation: objective 1.700000e+01, 316 iterations, 0.04 seconds

  Nodes |   Current Node |   Objective Bounds |   Work
Expl Unexpl |   Obj   Depth IntInf | Incumbent   BestBd   Gap | It/Node Time
*    0     0           0   17.000000   17.0000   0.00%   -    0s

Explored 0 nodes (316 simplex iterations) in 0.67 seconds
Thread count was 4 (of 4 available processors)

Solution count 2: 17 13

Optimal solution found (tolerance 1.00e-04)
Best objective 1.700000000000e+01, best bound 1.700000000000e+01, gap 0.0000%
74157

x[35518] = 1
x[52303] = 1
x[59231] = 1
x[73536] = 1
```

Figure 35: Screenshot of Optimal ILP Solution for Dataset  $D_3$

The optimal solution returned by Gurobi Optimiser selects cycles  $c_{35518}$ ,  $c_{52303}$ ,  $c_{59231}$  and  $c_{73536}$ , which are as follows:

$$c_{35518} = (0, 14, 15, 17, 8, 13, 18, 1, 0), c_{52303} = (3, 6, 20, 3), c_{59231} = (5, 10, 16, 5), \\ c_{73536} = (7, 11, 19, 7)$$

The optimal solution has a cardinality of 17, consisting of three 3-cycles and an ADC of length 8. As part of this solution, 15 patients from the matching pool and 2 patients from the active waiting list are allocated a kidney. The solution had a total run time of 2m 23s.

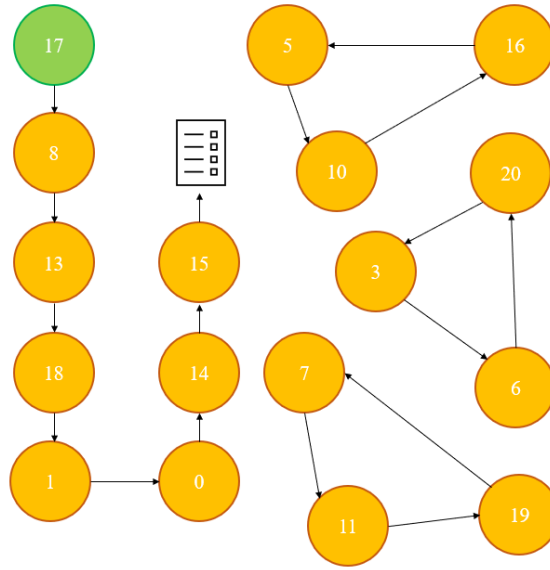


Figure 36: Optimal Solution Approximation for Dataset D<sub>3</sub>

Finally, ILP was used to compute an approximation to an optimal solution where the cycle length is unrestricted, but the cycle length search is restricted to a maximum of 10. In this instance, 76,064 cycles were identified.

Therefore, let  $C$  be represented by a  $76064 \times 21$  matrix.

As before, let a  $21 \times 1$  vector  $b$  represent the upper bound for each donor and let a  $21 \times 1$  vector  $x$  represent the selected cycles such that  $x_i = 1$  if and only if  $c_i$  belongs to the optimal solution.

Let a  $76064 \times 1$  vector  $k$  represent the length of each of the identified cycles.

Now, maximise  $kx$ , with respect to  $Ax \leq b$  and  $x_i \in \{0, 1\}$ .

When using the Gurobi Optimiser, the following output is returned:

```

2      61  3.9209961e+09 -9.2792778e+01 1s
3     100  3.5989964e+09 -9.2767998e+01 1s
4     138  3.1609968e+09 -9.2764682e+01 1s
5     175  2.6239974e+09 -9.2781817e+01 1s
6     213  2.0389979e+09 -9.2836295e+01 1s
7     252  1.3299987e+09 -9.2896609e+01 1s
8     289  7.4999924e+08 -9.2992636e+01 1s
9     326  1.5299983e+08 -9.3016016e+01 1s
10    373 -1.5003553e+01 -2.3669112e+01 1s
11    416 -1.7002368e+01 -2.3662022e+01 1s
12    464 -1.7002368e+01 -1.7005043e+01 1s

Sifting complete

Root relaxation: objective 1.700000e+01, 489 iterations, 0.08 seconds

   Nodes      |   Current Node   |   Objective Bounds   |   Work
  Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap   | It/Node Time
-----
    0     0    17.00000    0   8   14.00000   17.00000   21.4%   -    0s
H    0     0    16.00000    0   8   16.00000   17.00000    6.25%   -    0s
    0     0    17.00000    0   8   16.00000   17.00000    6.25%   -    1s
H    0     0    17.00000    0   8   17.00000   17.00000    0.00%   -    1s
    0     0    17.00000    0   8   17.00000   17.00000    0.00%   -    1s

Explored 1 nodes (489 simplex iterations) in 1.03 seconds
Thread count was 4 (of 4 available processors)

Solution count 4: 17 16 14 13

Optimal solution found (tolerance 1.00e-04)
Best objective 1.700000000000e+01, best bound 1.700000000000e+01, gap 0.0000%
76064

x[12875] = 1
x[38527] = 1
x[75737] = 1

```

Figure 37: Screenshot of Output for ILP Dataset D<sub>3</sub>

The optimal solution selected cycles  $c_{12875}$ ,  $c_{38527}$  and  $c_{75737}$ , which are as follows:

$$c_{12875} = (0, 4, 17, 9, 13, 6, 0), c_{38527} = (1, 2, 16, 20, 14, 11, 18, 1), c_{75737} = (8, 10, 15, 19, 8)$$

The solution had a run time of 2m 46s. It has a cardinality of 17, consisting of 15 patients from the matching pool and 2 from the active waiting list. The cardinality of this solution matches the cardinality where the cycle length was restricted. Therefore, this solution does not offer an improvement, and shorter cycle lengths are preferable due to reduced risk of transplants falling through post-matching.

#### 4.4.1 Top Trading Cycle Results Dataset 1

The top trading cycle generates a list of preferred donors for each patient, from most compatible match to least compatible match. The initial preference list for each patient is shown in Table 7 below:

Patient	Matching Donors			
	1st	2nd	3rd	4th, ...
0	6	1		
1	6			
2	1	6		
3	1	0		
4	1	0	6	
5	1	0	6	
6	0	4	3	1, 13
7	6	0	1	
8	6			
9	1			
10	5	6	8	9
11	14	12	10	9, 8, 7, 5, 1, 6, 0
12	6	1	0	
13	8	7	1	9, 5, 0
14	0			
15	6	5	0	14, 12, 10, 9, 7, 1

Table 7: Donor Preferences for each Patient in Dataset D<sub>1</sub>

As demonstrated by Table 7 above, donors  $d_2$ ,  $d_{11}$ , and  $d_{15}$  are not a match to any patients in the pool, therefore the corresponding patient-donor pairs are also removed from the pool. As a result, there are 13 patients remaining. The cycle length  $k$  is restricted such that  $2 \leq k \leq 3$ .

The optimal solution returned is the following cycle:

$$(1, 6, 0, 1)$$

```

Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6], 3: [1, 0], 4: [1, 0, 6], 5: [1, 0, 6], 6: [0, 4, 3, 1, 13], 7: [6, 0, 1], 8: [6], 9: [1], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0], 13: [8, 7, 1, 9, 5, 0], 14: [0], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1]}
Patients matched: 3
Optimal solution: {1: 6, 6: 0, 0: 1}

```

Figure 38: Screenshot of Optimal TTC Solution for Dataset D<sub>1</sub>, Cycle Length Restricted.

This cycle has a cardinality of 3, matching the maximum cardinality identified by ILP. The cycle chosen as optimal by TTC differs from ILP as TTC takes the weight of each edge into consideration. Therefore, the solution identified by TTC is more optimal as the likelihood of the transplant taking place and achieving successful results is higher.

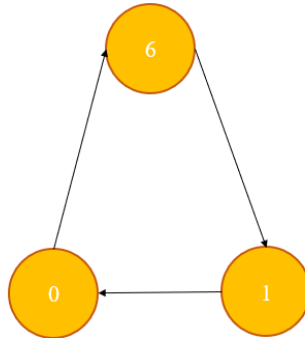


Figure 39: Optimal TTC Solution for Dataset D<sub>1</sub>

Additionally, TTC was used to identify the maximum cardinality solution where cycle length  $k$  was unrestricted. The initial preference list remains the same as in the example above.

The optimal solution returned the cycle shown below:

(5, 1, 6, 13, 5)

```

Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6], 3: [1, 0], 4: [1, 0, 6], 5: [1, 0, 6], 6: [0, 4, 3, 1, 13], 7: [6, 0, 1], 8: [6], 9: [1], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0], 13: [8, 7, 1, 9, 5, 0], 14: [0], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1]}
Patients matched: 4
Optimal solution: {5: 1, 1: 6, 6: 13, 13: 5}

```

Figure 40: Screenshot of TTC Solution for D<sub>1</sub>, Cycle Length Unrestricted

The implementation of TTC includes some additional definitions of optimality to investigate the effect on the optimal solution. The first extension for this dataset, is to prioritise the patients who have the most accelerated cases of CKD, namely stage 5 or end stage kidney failure.

The optimal solution, inclusive of the stage of each patient's condition, returned the following cycle:

$$(7, 6, 13, 7)$$

```

Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6], 3: [1, 0], 4: [1, 0, 6], 5: [1, 0, 6], 6: [0, 4, 3, 1, 13], 7: [6, 0, 1], 8: [6], 9: [1], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0], 13: [8, 7, 1, 9, 5, 0], 14: [0], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1]}
Stage 5 patients matched: 2
Patients matched: 3
Optimal solution: {7: 6, 6: 13, 13: 7}

```

Figure 41: Screenshot of Optimal TTC Solution with Stage 5 Priority

In this instance, the patients included in the original optimal solution are not end-stage CKD patients. Therefore, when the stage of each patient's condition was considered in the matching, the optimal matching has the same cardinality but the patients included are different.

A second extension of the TTC algorithm considers which patients are difficult to match with respect to sensitisation levels. In this dataset, the patients with the highest levels of sensitisation  $p_2$  and  $p_{10}$  have already been removed. Therefore, from the remaining patients, the patients with the highest levels of sensitisation are  $p_1$ ,  $p_3$ ,  $p_9$  and  $p_{11}$  at 0.45.

The returned optimal solution inclusive of high levels of sensitisation is as follows:

$$(1, 6, 0, 1)$$

```

Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6], 3: [1, 0], 4: [1, 0, 6], 5: [1, 0, 6], 6: [0, 4, 3, 1, 13], 7: [6, 0, 1], 8: [6], 9: [1], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0], 13: [8, 7, 1, 9, 5, 0], 14: [0], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1]}
Patients matched: 3
Optimal solution: {1: 6, 6: 0, 0: 1}
Patient: 1 Sensitisation: 0.9
Patient: 6 Sensitisation: 0.2875
Patient: 0 Sensitisation: 0.45

```

Figure 42: Screenshot of Optimal TTC Solution with Sensitisation Priority

The levels of sensitisation for patients  $p_0$ ,  $p_1$  and  $p_6$  are 0.45, 0.9 and 0.2875, respectively.

#### 4.4.2 Top Trading Cycle Results Dataset 2

As in the example above, the initial preference list for each patient is generated and shown in Table 8:



Matching Donors				
Patient	1st	2nd	3rd	4th, ...
0	6	1		
1	6			
2	1	6	17	16
3	1	17	0	
4	1	0	17	6, 16
5	1	0	17	16, 6
6	0	4	3	1, 13, 17
7	6	0	16	17, 1
8	16	6	17	
9	1	17		
10	5	6	8	9
11	14	12	10	9, 8, 7, 5, 1, 6, 0
12	6	1	0	17, 17
13	8	7	1	9, 5, 0
14	17	0	16	
15	6	5	0	14, 12, 10, 9, 7, 1
16	1	0	3	15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2
17	1	0	4	3, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5

Table 8: Donor Preferences for each Patient in Dataset D<sub>2</sub>

The dummy patients  $p_{16}$  and  $p_{17}$  associated with the altruistic donors  $d_{16}$  and  $d_{17}$  respectively, are compatible with each of the donors in the pool. As a result, no patient-donor pairs are removed from the pool due to lack of preference of a donor from another patient. This means that patient-donor pairs  $p_2d_2$ ,  $p_{11}d_{11}$  and  $p_{15}d_{15}$  remain in the pool for potential matching. As a result, all 17 patients included in the dataset remain in the pool, an addition of 3 excluding the altruistic donors on dataset D<sub>1</sub>. The cycle length  $k$  is restricted such that  $2 \leq k \leq 3$ .

The optimal solution returned is the following cycles:

(4, 1, 6, 4), (2, 17, 2), (5, 16, 15, 5)

```
C:\Users\molly\OneDrive\Documents\year 5\python\topn_matchingcycle.py
Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6, 17, 16], 3: [1, 17, 0], 4: [1, 0,
17, 6, 16], 5: [1, 0, 17, 16, 6], 6: [0, 4, 3, 1, 13, 17], 7: [6, 0, 16, 17, 1], 8
: [16, 6, 17], 9: [1, 17], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0],
12: [6, 1, 0, 17, 16], 13: [8, 7, 1, 9, 5, 0], 14: [17, 0, 16], 15: [6, 5, 0, 14,
12, 10, 9, 7, 1], 16: [1, 0, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [
1, 0, 4, 3, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5]}
Patients matched: 8
Optimal solution: {4: 1, 1: 6, 6: 4, 2: 17, 17: 2, 5: 16, 16: 15, 15: 5}
```

Figure 43: Screenshot of Optimal TTC Solution for Dataset D<sub>2</sub> with Cycle Length Restrictions

This solution consists of two 3-cycles and a short domino chain. The cardinality of this solution is 8, improving on the previous TTC results, which do not include altruistic donors, by 5 patients. In comparison, the ILP formulation identified a solution with cardinality 9, meaning an additional 2 patients could be matched.

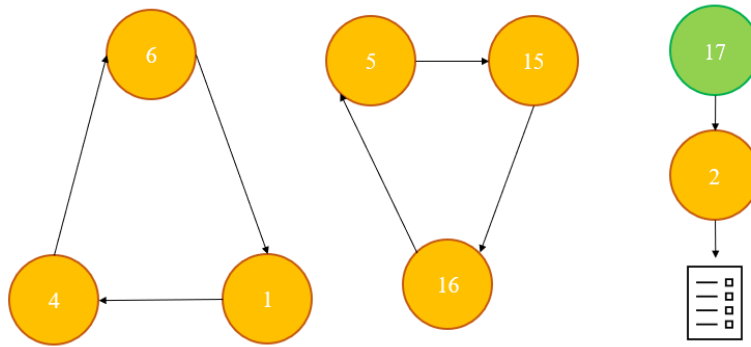


Figure 44: Optimal TTC Solution for Dataset D<sub>2</sub>

Again, TTC was used to identify the maximum cardinality solution where cycle length  $k$  was unrestricted. The initial preference list remains the same as in the examples above.

The optimal solution returned the cycles shown below:

(2, 1, 6, 4, 17, 2), (5, 16, 15, 5)

```
Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6, 17, 16], 3: [1, 17, 0], 4: [1, 0,
17, 6, 16], 5: [1, 0, 17, 16, 6], 6: [0, 4, 3, 1, 13, 17], 7: [6, 0, 16, 17, 1], 8
: [16, 6, 17], 9: [1, 17], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0],
12: [6, 1, 0, 17, 16], 13: [8, 7, 1, 9, 5, 0], 14: [17, 0, 16], 15: [6, 5, 0, 14,
12, 10, 9, 7, 1], 16: [1, 0, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [
1, 0, 4, 3, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5]}
Patients matched: 8
Optimal solution: {2: 1, 1: 6, 6: 4, 4: 17, 17: 2, 5: 16, 16: 15, 15: 5}
```

Figure 45: Screenshot of Optimal TTC Solution for Dataset D<sub>2</sub>, No Cycle Length Restrictions

The optimal solution has a cardinality of 8. This is only less than the cardinality of the solution identified by ILP, which had a cardinality of 10. One potential explanation for this is that the order in which patient-donor pairs are selected in TTC is restricted to choosing the most compatible donor who is still available.

One extension used with this dataset is the priority of patients who have end-stage kidney disease. The optimal solution, inclusive of the stage of each patient's condition, returned the following cycle:

$$(4, 1, 6, 4), (2, 17, 2), (5, 16, 15, 5)$$

```
Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6, 17, 16], 3: [1, 17, 0], 4: [1, 0, 17, 6, 16], 5: [1, 0, 17, 16, 6], 6: [0, 4, 3, 1, 13, 17], 7: [6, 0, 16, 17, 1], 8: [16, 6, 17], 9: [1, 17], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0, 17, 16], 13: [8, 7, 1, 9, 5, 0], 14: [17, 0, 16], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1], 16: [1, 0, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5]}
Stage 5 patients matched: 6
Patients matched: 8
Optimal solution: {4: 1, 1: 6, 6: 4, 2: 17, 17: 2, 5: 16, 16: 15, 15: 5}
```

Figure 46: Screenshot of Optimal TTC Solution for Dataset D<sub>2</sub> with Stage 5 Priority

Again, in this instance, 6 of the 6 patients included in the original optimal solution are end-stage CKD patients. Therefore, when considering the stage of each patient's condition in the matching, the included patients remain the same. However, this solution includes two 3-cycles and one short ADC.

Unlike dataset D<sub>1</sub>, due to the addition of two altruistic donors, the patients with the highest levels of sensitisation remain in the pool. Patients p<sub>1</sub> and p<sub>10</sub> both have levels of sensitisation at 0.9.

The returned optimal solution inclusive of high levels of sensitisation is as follows:

$$(2, 1, 6, 4, 17, 2), (5, 16, 15, 5)$$

```
Initial Preferences {0: [6, 1], 1: [6], 2: [1, 6, 17, 16], 3: [1, 17, 0], 4: [1, 0, 17, 6, 16], 5: [1, 0, 17, 16, 6], 6: [0, 4, 3, 1, 13, 17], 7: [6, 0, 16, 17, 1], 8: [16, 6, 17], 9: [1, 17], 10: [5, 6, 8, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 6, 0], 12: [6, 1, 0, 17, 16], 13: [8, 7, 1, 9, 5, 0], 14: [17, 0, 16], 15: [6, 5, 0, 14, 12, 10, 9, 7, 1], 16: [1, 0, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5]}
Patients matched: 8
Optimal solution: {4: 1, 1: 6, 6: 4, 2: 17, 17: 2, 5: 16, 16: 15, 15: 5}
Patient: 4 Sensitisation: 0.05
Patient: 1 Sensitisation: 0.9
Patient: 6 Sensitisation: 0.2875
Patient: 2 Sensitisation: 0.45
Patient: 17 Sensitisation: 0.05
Patient: 5 Sensitisation: 0.05
Patient: 16 Sensitisation: 0.05
Patient: 15 Sensitisation: 0.05
```

Figure 47: Screenshot of Optimal TTC Solution for Dataset D<sub>2</sub> with Sensitisation Priority

The sensitisation levels for patients p<sub>4</sub>, p<sub>1</sub>, p<sub>6</sub>, p<sub>2</sub>, p<sub>17</sub>, p<sub>5</sub>, p<sub>16</sub> and p<sub>15</sub> are 0.05, 0.9, 0.2875, 0.45, 0.05, 0.05, 0.05 and 0.05, respectively. The dummy patients p<sub>16</sub>, and

$p_{17}$  are assigned a low level of sensitisation as the actual recipients for donor  $d_2$  and  $d_5$  kidney have not yet been chosen.

#### 4.3.3 Top Trading Cycle Results Dataset 3

As in the example above, the initial preference list is generated and shown below:

Patient	Matching Donors			
	1st	2nd	3rd	4th, ...
0	6	1	18	19, 20
1	6	19	18	20
2	1	6	17	16, 20
3	1	20	17	0
4	1	0	17	6, 16, 20
5	1	0	17	16, 6, 20, 19
6	0	4	3	1, 19, 18, 13, 20, 17
7	6	0	16	17, 1, 20, 19
8	16	6	20	19, 17
9	20	1	19	17
10	5	6	8	20, 19, 9
11	14	12	10	9, 8, 7, 5, 1, 20, 6, 0
12	6	1	0	17, 16, 20, 19
13	8	7	1	20, 9, 5, 0
14	17	0	16	20, 19
15	6	5	0	20, 14, 12, 10, 9, 7, 1
16	1	0	20	3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2
17	1	0	4	3, 20, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5
18	1	10	5	4, 3, 20, 19, 17, 15, 14, 13, 12, 9, 6, 2, 0, 18, 16, 11, 8, 7
19	1	15	11	19, 17, 10, 20
20	20	16	6	

Table 9: Donor Preferences for each Patient in Dataset D<sub>3</sub>

Note that for the compatible patient-donor pairs  $p_{18}d_{18}$ ,  $p_{19}d_{19}$ ,  $p_{20}d_{20}$  the donor is listed as one of the patient's preferences, shown in bold in Table 9.

The optimal solution returned is the following cycles:

$$(0, 1, 6, 0), (5, 19, 15, 5), 18$$

```
Initial Preferences {0: [6, 1, 18, 19, 20], 1: [6, 19, 18, 20], 2: [1, 6, 17, 16, 20], 3: [1, 20, 17, 0], 4: [1, 0, 17, 6, 16, 20], 5: [1, 0, 17, 16, 6, 20, 19], 6: [0, 4, 3, 1, 19, 18, 13, 20, 17], 7: [6, 0, 16, 17, 1, 20, 19], 8: [16, 6, 20, 19, 17], 9: [20, 1, 19, 17], 10: [5, 6, 8, 20, 19, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 20, 6, 0], 12: [6, 1, 0, 17, 16, 20, 19], 13: [8, 7, 1, 20, 9, 5, 0], 14: [17, 0, 16, 20, 19], 15: [6, 5, 0, 20, 14, 12, 10, 9, 7, 1], 16: [1, 0, 20, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 20, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5], 18: [1, 10, 5, 4, 3, 20, 19, 17, 15, 14, 13, 12, 9, 6, 2, 0, 18, 16, 11, 8, 7], 19: [1, 15, 11, 19, 17, 10, 20], 20: [20, 16, 6]}
```

Patients matched: 7  
Optimal solution: {1: 6, 6: 0, 0: 1, 5: 19, 19: 15, 15: 5, 18: 18}

Figure 48: Screenshot of Optimal TTC Solution for Dataset D<sub>3</sub>, Cycle Length Restricted

This solution contains two 3-cycles, and one compatible patient-donor matching. The cardinality of this solution is 7. As patient  $p_{20}$  was not matched as part of the solution, it is possible for them to be matched with donor  $d_{20}$ . A limitation of TTC in this case is that the  $p_{20} \rightarrow d_{20}$  matching was not identified for this solution due to the order in which the preferences are reduced. In addition, the altruistic donors are not utilised, indicating this solution could be optimised further still.

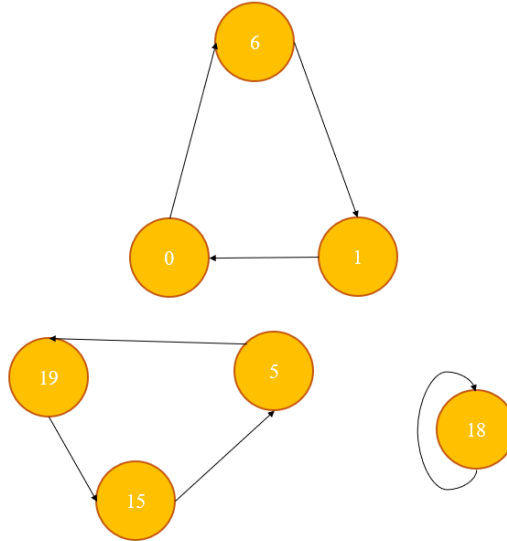


Figure 49: Optimal TTC Solution for Dataset D<sub>3</sub>

Again, TTC was used to identify the maximum cardinality solution where cycle length  $k$  was unrestricted. The initial preference list remains the same as in the example above.

The optimal solution returned the cycles shown below:

$$(4, 1, 6, 0, 18, 10, 5, 17, 4), (7, 19, 15, 7)$$

```

Initial Preferences {0: [6, 1, 18, 19, 20], 1: [6, 19, 18, 20], 2: [1, 6, 17, 16, 20], 3: [1, 20, 17, 0], 4: [1, 0, 17, 6, 16, 20], 5: [1, 0, 17, 16, 6, 20, 19], 6: [0, 4, 3, 1, 19, 18, 13, 20, 17], 7: [6, 0, 16, 17, 1, 20, 19], 8: [16, 6, 20, 19, 17], 9: [20, 1, 19, 17], 10: [5, 6, 8, 20, 19, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 20, 6, 0], 12: [6, 1, 0, 17, 16, 20, 19], 13: [8, 7, 1, 20, 9, 5, 0], 14: [17, 0, 16, 20, 19], 15: [6, 5, 0, 20, 14, 12, 10, 9, 7, 1], 16: [1, 0, 20, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 20, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5], 18: [1, 10, 5, 4, 3, 20, 19, 17, 15, 14, 13, 12, 9, 6, 2, 0, 18, 16, 11, 8, 7], 19: [1, 15, 11, 19, 17, 10, 20], 20: [20, 16, 6]}
Patients matched: 11
Optimal solution: {4: 1, 1: 6, 6: 0, 0: 18, 18: 10, 10: 5, 5: 17, 17: 4, 7: 19, 19: 15, 15: 7}

```

Figure 50: Screenshot of Optimal TTC Solution for Dataset D<sub>3</sub>, Cycle Length Unrestricted

The optimal solution has a cardinality of 11.

One extension used with this dataset is the priority of patients who have end-stage kidney disease. The optimal solution, inclusive of the stage of each patient's condition, returned the following cycle:

(0, 1, 6, 0), (5, 19, 10, 5), 18

```

Initial Preferences {0: [6, 1, 18, 19, 20], 1: [6, 19, 18, 20], 2: [1, 6, 17, 16, 20], 3: [1, 20, 17, 0], 4: [1, 0, 17, 6, 16, 20], 5: [1, 0, 17, 16, 6, 20, 19], 6: [0, 4, 3, 1, 19, 18, 13, 20, 17], 7: [6, 0, 16, 17, 1, 20, 19], 8: [16, 6, 20, 19, 17], 9: [20, 1, 19, 17], 10: [5, 6, 8, 20, 19, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 20, 6, 0], 12: [6, 1, 0, 17, 16, 20, 19], 13: [8, 7, 1, 20, 9, 5, 0], 14: [17, 0, 16, 20, 19], 15: [6, 5, 0, 20, 14, 12, 10, 9, 7, 1], 16: [1, 0, 20, 3, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 20, 2, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5], 18: [1, 10, 5, 4, 3, 20, 19, 17, 15, 14, 13, 12, 9, 6, 2, 0, 18, 16, 11, 8, 7], 19: [1, 15, 11, 19, 17, 10, 20], 20: [20, 16, 6]}
Stage 5 patients matched: 5
Patients matched: 7
Optimal solution: {1: 6, 6: 0, 0: 1, 5: 19, 19: 15, 15: 5, 18: 18}

```

Figure 51: Screenshot of Optimal TTC Solution for Dataset D<sub>3</sub> with Stage 5 Priority

In this instance, 5 of the 7 patients included in the original optimal solution are end-stage CKD patients. Therefore, when considering the stage of each patient's condition in the matching, the optimal solution remains the same.

The returned optimal solution inclusive of high levels of sensitisation is as follows:

(0, 1, 6, 0), (2, 16, 2), (3, 17, 3), (5, 19, 10, 5), 18

```

Initial Preferences {0: [6, 1, 18, 19, 20], 1: [6, 19, 18, 20], 2: [1, 6, 17, 16, 2
0], 3: [1, 20, 17, 0], 4: [1, 0, 17, 6, 16, 20], 5: [1, 0, 17, 16, 6, 20, 19], 6: [
0, 4, 3, 1, 19, 18, 13, 20, 17], 7: [6, 0, 16, 17, 1, 20, 19], 8: [16, 6, 20, 19, 1
7], 9: [20, 1, 19, 17], 10: [5, 6, 8, 20, 19, 9], 11: [14, 12, 10, 9, 8, 7, 5, 1, 2
0, 6, 0], 12: [6, 1, 0, 17, 16, 20, 19], 13: [8, 7, 1, 20, 9, 5, 0], 14: [17, 0, 16
, 20, 19], 15: [6, 5, 0, 20, 14, 12, 10, 9, 7, 1], 16: [1, 0, 20, 3, 15, 14, 13, 12
, 11, 10, 9, 8, 7, 6, 5, 4, 2], 17: [1, 0, 4, 3, 20, 2, 15, 14, 13, 12, 11, 10, 9,
8, 7, 6, 5], 18: [1, 10, 5, 4, 3, 20, 19, 17, 15, 14, 13, 12, 9, 6, 2, 0, 18, 16, 1
1, 8, 7], 19: [1, 15, 11, 19, 17, 10, 20], 20: [20, 16, 6]}
Patients matched: 7
Optimal solution: {1: 6, 6: 0, 0: 1, 5: 19, 19: 15, 15: 5, 18: 18}
Patient: 1 Sensitisation: 0.9
Patient: 6 Sensitisation: 0.2875
Patient: 0 Sensitisation: 0.45
Patient: 5 Sensitisation: 0.05
Patient: 19 Sensitisation: 0.05
Patient: 15 Sensitisation: 0.05
Patient: 18 Sensitisation: 0.05

```

Figure 52: Screenshot of Optimal TTC Solution for Dataset D<sub>3</sub> with Sensitisation Priority

The sensitisation levels for patients p<sub>1</sub>, p<sub>6</sub>, p<sub>0</sub>, p<sub>5</sub>, p<sub>19</sub>, p<sub>15</sub>, p<sub>18</sub> are 0.9, 0.2875, 0.45, 0.05, 0.05, 0.05, 0.05 and 0.05, respectively.

#### 4.4 Overall Results Analysis

k = cycle length, l = chain length

##### Dataset 1

There are 16 patients in this dataset. There are no altruistic donors or compatible patient-donor pairs.

Method	Edges	Constraints	Cardinality	Additional Information
Pairwise	3	$k = 2$	2	
ILP	28	$2 \leq k \leq 3, 2 \leq l \leq 3$	3	
TTC	56	$2 \leq k \leq 3, 2 \leq l \leq 3$	3	
ILP	28	$2 \leq k \leq 3$	3	
TTC, priority stage 5	56	$2 \leq k \leq 3, 2 \leq l \leq 3$	3	2 Stage 5 patients matched
TTC, priority sensitisation	56	$2 \leq k \leq 3, 2 \leq l \leq 3$	3	Sensitisation of patients matched: 0.45, 0.9 and 0.2875
ILP	28	None	5	
TTC	56	None	5	

Table 10: Results Table for Dataset D<sub>1</sub>

From this dataset, it is clear that pairwise exchanges offered the least optimal solution as it holds the lowest cardinality solution within the set. TTC and ILP computed solutions with the same cardinality for each set of constraints. Priority of stage 5 patients and patients with high levels of sensitisation did not impact the cardinality of the optimal solution.

## Dataset 2

This dataset contains 18 patient-donor pairs, including 2 altruistic donors.

Method	Edges	Constraints	Cardinality	Additional Information
Pairwise	21	$k = 2$	6	
ILP	105	$2 \leq k \leq 3, 2 \leq l \leq 3$	9	
TTC	105	$2 \leq k \leq 3, 2 \leq l \leq 3$	8	
ILP	105	$2 \leq k \leq 3$	12	
TTC, priority stage 5	105	$2 \leq k \leq 3, 2 \leq l \leq 3$	8	6 Stage 5 patients matched
TTC, priority sensitisation	105	$2 \leq k \leq 3, 2 \leq l \leq 3$	8	Sensitisation of patients matched: 0.05, 0.9, 0.2875, 0.45, 0.05, 0.05, 0.05 and 0.05
ILP	105	None	12	
TTC	105	None	8	

Table 11: Results Table for Dataset D<sub>2</sub>

The results from dataset D<sub>2</sub> support the conclusion from dataset D<sub>1</sub> that the pairwise exchange allocation offers an inferior solution in comparison with TTC and ILP. However, these results differ from the conclusions drawn from the previous dataset as TTC fails to find a solution with a cardinality that matches that of ILP. In this instance, the optimal solution identified by ILP has a cardinality that is 5 greater than that of TTC. As a result, the ILP formulation offers the most optimal solution in this case. As before, prioritising stage 5 patients or patients with high levels of sensitisation did not lower the maximum cardinality of the solution.



### Dataset 3

This dataset contains 21 patient-donor pairs, including 2 altruistic donors and 3 compatible patient-donor pairs.

Method	Edges	Constraints	Cardinality	Additional Information
Pairwise	57	$k = 2$	10	N/A
ILP	167	$2 \leq k \leq 3,$ $2 \leq l \leq 3$	15	N/A
TTC	167	$2 \leq k \leq 3, 2 \leq l \leq 3$	7	
ILP	167	$2 \leq k \leq 3$	17	
TTC, priority stage 5	167	$2 \leq k \leq 3,$ $2 \leq l \leq 3$	7	5 Stage 5 patients matched
TTC, priority sensitisation	167	$2 \leq k \leq 3,$ $2 \leq l \leq 3$	7	Sensitisation of patients matched: 0.9, 0.2875, 0.45, 0.05, 0.05, 0.05, 0.05 and 0.05
ILP	167	None	17	
TTC	167	None	11	

Table 12: Results Table for Dataset D<sub>3</sub>

The largest dataset D<sub>3</sub>, and the dataset including the most diverse range of patient-donor pairs, confirms the results offered by dataset D<sub>2</sub>. This means that the optimal solution offered by the pairwise exchange had a maximum cardinality of 10, 7 kidney allocations less than the maximum cardinality for the dataset. Again, ILP identified the solution with the greatest cardinality, even though its performance was compromised to an approximation for this dataset.

Dataset D<sub>1</sub> represents a basic dataset where every patient-donor pair in the matching pool are not compatible with each other. Out of the 16 patients in the pool, the maximum matching only offered kidney allocations for 3 patients. This means that only 18.75% of patients were able to receive a kidney and 81.25% of the available kidneys were wasted.

In comparison, dataset  $D_2$  contained the same patients as dataset  $D_1$ , with the addition of two altruistic donors to highlight the importance of altruistic donors in the matching pool. The two altruistic donors were included in every optimal matching, emphasising their importance to the matching pool. The optimal solution for this dataset had a cardinality of 12. This means that the addition of just two altruistic donors facilitated a further 9 matches. In this case, 75% of patients in the pool were able to receive a kidney, alongside 2 patients from the active waiting list and 33% of the available kidneys were wasted. This represents a 53.8% decrease in waste of available donors.

Finally, dataset  $D_3$  contained the same patients as dataset  $D_2$ , with the addition of three compatible patient-donor pairs. This aims to expose the importance of introducing compatible patient-donor pairs into the matching pool, a technique that is not currently practised in the UK under the NHS. This technique guarantees at least as many matches as the pool which excludes compatible pairs as where the additional pairs are not matched as part of the optimal solution, they are matched to one another. The optimal solution for this dataset had a cardinality of 17. This means that with three additional pairs, five additional kidney allocations were made. In addition, in the solution offered by TTC, only one of the compatible pairs were matched to themselves, indicating that the other pairs were able to achieve a better-matched kidney than the one they entered the pool with. This result achieves an allocation for 89.47% of patients in the matching pool, with 2 patients from the waiting list also receiving kidneys. As a result, only 2 available kidneys were unable to be allocated as part of the matching pool, indicating only 9.52% of donors were wasted.

## 5 Future Work

One area where progress could be made to improve on the research and methods detailed within this project would be the development of a front-end application to display and demonstrate each of the graphs, methods, and results. During the initial planning of this project, one of the aims was to develop a front-end application to visualise the data for a user, to provide graphics to aid understanding of the processes within each of the methods and algorithms, and finally to display the resulting matchings and analysis clearly and visually. This application could offer multiple datasets for the user to choose from, and apply the methods researched as part of this project. A further feature of this application could be for the user to be able to input and build datasets or graphs and compute various optimal solutions. Although this application would not impact the results provided by the methods offered as part of this project, it would be a valuable addition for a user. This is because researching, practising, and understanding the algorithms included within the project can be challenging and time-consuming and therefore it can be difficult for users to understand the steps between choosing a dataset and viewing the results. A graphic that outlines the steps of each algorithm, manipulating the graph as the algorithm progresses, would allow a user of any background to gain an understanding of each method. Additionally, viewing the resulting cycles and chains for each matching on the graph allowed me to gain a deeper understanding of the solution.

A second area for future development is to further explore the suggestion of a multi-national kidney exchange more closely. To draw on one of the conclusions of this project, more transplants are facilitated when compatible patient-donor pairs are entered into the matching pool. One idea which stems from this is that more transplant allocations are made when transplant and waiting list information is transparent. This means that allocating compatible patient-donor pairs to each other without considering other options limits the opportunities for pairs in the matching pool and limits the compatible pair from potentially identifying a more compatible match. The application of this idea internationally implies that more transplants could be facilitated by sharing transplant and waiting list information internationally.

## 6 Conclusions

As this project concludes, it is important to consider the aims and to what extent they were achieved. To reiterate, this project aimed to create a tool to visualise the performance of different optimisation methods and analyse the performance of the chosen methods and consider their strengths and weaknesses. Early in this project, the focus began to shift from the creation of an application to visualise each of the chosen methods. This is because the definition of optimality applied to kidney exchange solutions, and the impact of different types of living donors such as altruistic and compatible patient-donor pair donors quickly became the focus as initial research began. Therefore, the first aim of this project was not met but was exchanged for a different research path which arguably is more valuable.

The second aim, to analyse the performance of different optimisation techniques, is achieved as part of this project. The methods chosen and analysed as part of this project are Blossom's Algorithm, modelling the pairwise exchange, an ILP formulation that relies on Kosaraju's and Johnson's Algorithm, and the Top Trading Cycles and Chains method. To reiterate the conclusions drawn in Chapter 4, the ILP Formulation produced the optimal solutions by definition of greatest cardinality, despite only calculating an approximation to the optimal solution for the final dataset.

The additional aim of this project was to explore the impact of different donors on the cardinality of the optimal solution. This aim was achieved by extending the initial dataset to include altruistic donors and compatible patient-donor pairs and analyse the effects on the optimal solution. As discussed in Chapter 4, both altruistic donors and the introduction of compatible patient-donor pairs into the matching pool significantly increased the cardinality of the optimal solutions.

These results suggest that new models for kidney exchange programs should consider the prospect of offering compatible patient-donor pairs the opportunity to enter the matching pool. Patients may be encouraged to choose this option on the premise that it may facilitate more kidney allocations, and therefore allow them to help others in a similar position to themselves and provide the opportunity to receive a more compatible kidney.

Future work could address the initial aim of this project to produce a tool to visualise the processes taken to generate an optimal matching and provide a solid understanding to user interested in either the kidney exchange or the underlying algorithms.

To conclude, this project has successfully explored the performance of different optimisation techniques, provided discussions on the potential limitations, and identified the most appropriate method based on the results displayed in Chapter 4. In addition, the impact of altruistic donors and the introduction of compatible patient-donor pairs into the matching pool has been explored and conclusions provided to support ongoing research into the advancement of kidney exchange models.

## 7 Reflection on Learning

Whilst undertaking this project, I have developed my skills in independent research. This was achieved because I had to consider a broad range of topics, from mathematics and graph theory to biology, and the underlying processes behind kidney transplantation and matching. As part of this research, I have discovered and understood an array of complex algorithms, such as Kosaraju's Algorithm and Edmond's Blossom Algorithm, working independently to gain knowledge into the methods involved for each algorithm. A particular challenge was that many of these algorithms were reported only in mathematical research papers, which I was unfamiliar with at the beginning of this project. Additionally, I have also improved my technical skills whilst undertaking this project as I have learnt to use Gurobi Optimiser to formulate and solve an ILP problem with Python code. Similarly, I have gained experience implementing unfamiliar and complex algorithms in Python from pseudocode produced by myself or found in research papers. When selecting a dataset to model optimisation techniques on, I found the application of the kidney exchange fascinating, despite a lack of knowledge regarding the biology and technical matching process before starting this project. I have therefore developed and expanded my knowledge during this project in several areas which were previously unfamiliar to me.

Furthermore, I also utilised my time management skills to ensure the timely completion of the project to a high standard. Reflecting on my initial plan, I vastly underestimated the time it would take to research relevant algorithms, fully understand the methods within them and finally implement them and gather my results using a variety of datasets. Time management was therefore a vital skill before and during the project to ensure effective research, implementation and report writing.

Throughout this project, I have worked closely with my supervisor Dr Richard Booth. This has enabled me to develop my communication skills as I provided weekly updates on my progress, verbalised my ideas clearly, and requested advice where necessary. I also had to navigate the organisation of these meetings entirely virtually, which was achieved by video calling to ensure effective and regular communication. In my initial plan, I agreed to attend weekly meetings with Dr Booth lasting around 30 minutes, including two progress meetings, which were upheld throughout the entire project.

Undertaking this project has also helped me to develop my report writing skills as clear written communication detailing the relevant background information, methods and implementations is of utmost importance. In particular, providing sufficient detail whilst maintaining a clear, concise report writing style has been challenging. Additionally, my oral presentation skills have improved as I develop a clear and professional demonstration for my viva presentation.

Referring back to my initial plan, the project aims and deliverables presented were as follows:

In this project I aim to:

- Create a tool to visualise the performance of different algorithms and optimisation methods of kidney exchange assignments.
- Analyse the performance of each of the chosen methods and consider the strengths and weaknesses of each.

Project Objectives:

- Study the optimal matching problem, in particular its application to the kidney exchange.
  - Produce a mathematical description of the problem.
- Research and implement algorithms that provide optimal solutions or matchings – or at least a good approximation to an optimal matching.
  - Implement chosen algorithms and visualise the results via a user-friendly interface.
- Conduct analysis of the chosen algorithms or methods using the application of the kidney exchange.
  - Produce a report explaining the chosen algorithms and offer analysis of their performance.

Unfortunately, the final result of this project does not offer a user-friendly interface to visualise the resulting matchings of each optimisation technique. This is due to the vast amount of time required to research, understand, and implement each algorithm, developing a frontend alongside this seemed infeasible under the time constraints. However, the second project aim to analyse the performance and results of each algorithm has been met.

Concerning the project objectives, I believe I have achieved the overarching objectives to study the optimal matching problem and the kidney exchange, research and implement different methods and conduct analysis of the results for each.

In conclusion, completing the project has taught me a new style of working and researching independently, specifically learning from mathematical papers, and expressing what I have learnt both professionally and mathematically. This process, although more time consuming than first expected, has been very rewarding and I have thoroughly enjoyed the opportunity to research and implement techniques within optimisation with application to the kidney exchange.



## Table of Abbreviations

ADCs	Altruistic Donor Chains
BILP	Binary Linear Integer Programming
CKD	Chronic Kidney Disease
BDB	Donation after Brainstem Death
DCD	Donation after Cardiac Death
DFS	Depth First Search
HLA	Human Leukocyte Antigens
ILP	Integer Linear Programming
NDADs	Non-directed Altruistic Donors
NHS	National Health Service
PPD	Paired/Pooled Donation
SCC	Strongly Connected Component
TTC	Top Trading Cycle
UKLKSS	UK Living Kidney Sharing Scheme

## References

- [1] - Biró, P., Manlove, D. and Rizzi, R., 2009. Maximum Weight Cycle Packing in Directed Graphs, with Application to Kidney Exchange Programs. *Discrete Mathematics, Algorithms and Applications*, 01(04), pp.499-517.
- [2] - Biró, P., van de Klundert, J., Manlove, D., Pettersson, W., Andersson, T., Burnapp, L., Chromy, P., Delgado, P., Dworczak, P., Haase, B., Hemke, A., Johnson, R., Klimentova, X., Kuypers, D., Nanni Costa, A., Smeulders, B., Spieksma, F., Valentín, M. and Viana, A., 2021. Modelling and optimisation in European Kidney Exchange Programmes. *European Journal of Operational Research*, 291(2), pp.447-456.
- [3] - Edmonds, J., 1965. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17, pp.449-467.
- [4] - GeeksforGeeks. 2020. Strongly Connected Components - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/strongly-connected-components/>> [Accessed 3 May 2021].
- [5] - Giscard, P., Kriege, N. and Wilson, R., 2019. A General Purpose Algorithm for Counting Simple Cycles and Simple Paths of Any Length. *Algorithmica*, 81(7), pp.2716-2737.
- [6] - Hendry, R. and Robb, M., 2020. Annual Report on Kidney Transplantation. [online] Statistics and Clinical Studies, NHS Blood and Transplant, pp.2-24. Available at: <<https://nhsbtdbe.blob.core.windows.net/umbraco-assets-corp/20032/kidney-annual-report-2019-20-final.pdf>> [Accessed 12 April 2021].
- [7] - National Kidney Foundation. 2016. Incompatible Blood Types and Paired Exchange Programs. [online] Available at: <<https://www.kidney.org/atoz/content/incompatible-blood-types-and-paired-exchange-programs>> [Accessed 17 April 2021].
- [8] - Kerr, M., 2012. Chronic Kidney Disease in England: The Human and Financial Cost. [ebook] NHS Kidney Care by Insight Health Economics, pp.5-23. Available at: <<https://www.england.nhs.uk/improvement-hub/wp-content/uploads/sites/44/2017/11/Chronic-Kidney-Disease-in-England-The-Human-and-Financial-Cost.pdf>> [Accessed 8 April 2021].
- [9] - OpenGenus IQ: Learn Computer Science. 2021. Kosaraju's Algorithm for Strongly Connected Components  **$O(V+E)$** . [online] Available at: <<https://iq.opengenus.org/kosarajus-algorithm-for-strongly-connected-components>> [Accessed 9 April 2021].
- [10] - Lee, Y., Chang, J., Choi, H., Jung, J., Kim, Y., Chung, W., Park, Y. and Lee, H., 2012. Donor-Recipient Age Difference and Graft Survival in Living Donor Kidney Transplantation. *Transplantation Proceedings*, 44(1), pp.270-272.

- [11] - Mak-Hau, V., 2015. On the kidney exchange problem: cardinality constrained cycle and chain problems on directed graphs: a survey of integer programming approaches. *Journal of Combinatorial Optimization*, 33(1), pp.35-59.
- [12] - Manlove, D. and O'Malley, G., 2015. Paired and Altruistic Kidney Donation in the UK. *ACM Journal of Experimental Algorithmics*, 19(2), pp.1-21.
- [13] - Moore, K. and Landman, N., 2021. Blossom Algorithm | Brilliant Math & Science Wiki. [online] Brilliant.org. Available at: <<https://brilliant.org/wiki/blossom-algorithm/>> [Accessed 4 May 2021].
- [14] - ODT Clinical - NHS Blood and Transplant. 2021. UK Living Kidney Sharing Scheme. [online] Available at: <<https://www.odt.nhs.uk/living-donation/uk-living-kidney-sharing-scheme/>> [Accessed 6 April 2021].
- [15] - Roth, A., Sönmez, T. and Ünver, M., 2007. Efficient Kidney Exchange: Coincidence of Wants in Markets with Compatibility-Based Preferences. *American Economic Review*, 97(3), pp.828-851.
- [16] - Saidman, S., Roth, A., Sönmez, T., Ünver, M. and Delmonico, F., 2006. Increasing the Opportunity of Live Kidney Donation by Matching for Two- and Three-Way Exchanges. *Transplantation*, 81(5), pp.773-782.
- [17] - Shapley, L. and Scarf, H., 1974. On cores and indivisibility. *Journal of Mathematical Economics*, 1(1), pp.23-37.
- [18] - Shoemaker, A. and Vase, S., 2016. *Edmonds' Blossom Algorithm*. Stanford University, pp.4-7.
- [19] - Stewart, C., 2020. Patient deaths on organ transplant waiting list 2020 | Statista. [online] Statista. Available at: <<https://www.statista.com/statistics/519829/patient-deaths-on-organ-transplant-waiting-list-united-kingdom-uk/>> [Accessed 13 March 2021].
- [20] - Sullivan, R., 2020. Kidney transplants on the rise in UK thanks to change to sharing scheme. *Independent*, [online] Available at: <<https://www.independent.co.uk/news/uk/home-news/kidney-transplant-increase-nhs-altruistic-donors-sharing-a9293231.html>> [Accessed 23 April 2021].
- [21] - Programiz.com. 2021. Strongly Connected Components. [online] Available at: <<https://www.programiz.com/dsa/strongly-connected-components>> [Accessed 1 May 2021].