A comparison of matching algorithms for Kidney Exchange Programs

Tiago Monteiro, Xenia Klimentova, João Pedro Pedroso, Ana Viana
tiago.p.monteiro@inesctec.pt

INESC TEC, Porto, Portugal

December 10, 2018

Introduction

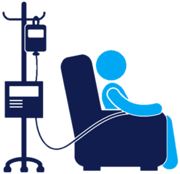
Kidney failure

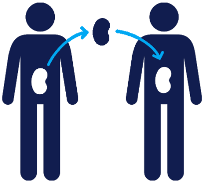
1 in 10 Europeans has some degree of chronic kidney disease (many people don't know it – silent disease)

500,000 people in the United States have **end-stage renal disease**.

There are **two treatment options** to stay alive:

- Dialysis
- **Transplantation**
  - Deceased donors
  - **Living donors**



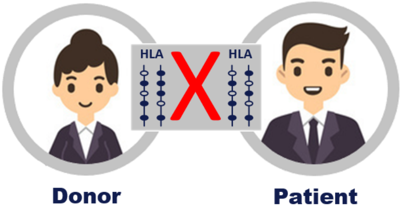


Kidney transplant – Incompatibility

- **Blood incompatibility**

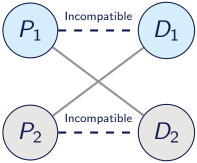
Donor	Patient			
	O	A	B	AB
O	√	√	√	√
A	X	√	X	√
B	X	X	√	√
AB	X	X	X	√

- **Immunological incompatibility**



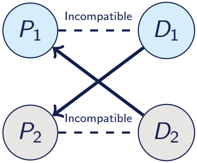
Kidney Exchange Programs

Kidney Exchange Programs - allow incompatible patient and living donor pairs, whose donor cannot provide a kidney to the respective patient, to have an “exchange” between them.



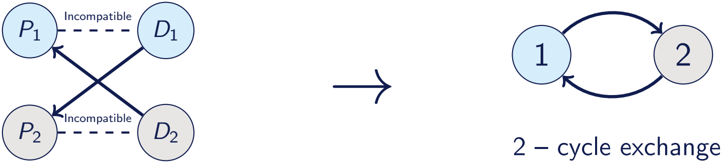
Kidney Exchange Programs

Kidney Exchange Programs - allow incompatible patient and living donor pairs, whose donor cannot provide a kidney to the respective patient, to have an "exchange" between them.



Kidney Exchange Programs

Kidney Exchange Programs - allow incompatible patient and living donor pairs, whose donor cannot provide a kidney to the respective patient, to have an "exchange" between them.



2 – cycle exchange

Graph representation:

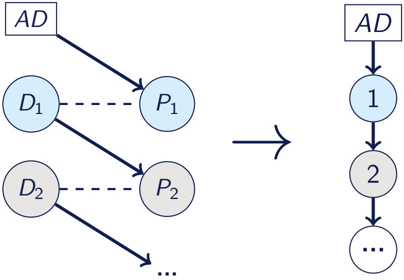
- **Vertices** represent incompatible patient-donor pairs
- **Arcs** represent compatibility between donor and patients

Kidney Exchange Programs

Altruistic donors – chain

Besides cycles, another possible organization for exchanges is a **chain** initiated by an **altruistic donor** (AD) – a donor with no associated patient that donates a kidney for no return.

- **AD** donates to P_1;
- D_1 donates to the patient in the following pair (P_2);
- ... and so forth ...



The last donor can donate either to the **deceased donors waiting list** or act as **bridge donor for the next matching**.

Kidney Exchange Programs

Cycle or chain bound

Cycle/chain size (k) **must be bounded:**

- all transplants in a cycle or chain must be done **simultaneously**.
    - causing logistic/personnel problems!
- planned exchanges may fail
    - due to last-minute incompatibilities;
    - patients/donors leave the program;
    - ...

Smaller cycles or chains – less pairs affected in case of failure!

Kidney Exchange Programs

Problem statement

Let $G(V, A)$ be a directed graph with:

V $= P \cup N$ – the set of vertices, composed by:

- P – set of incompatible patient-donor pairs;
- N – set of altruistic donors.

A – the set of arcs for representing compatibilities between the vertices.

Two vertices $i, j \in V$ are connected by arc (i, j) if the patient in vertex j is compatible with the donor in vertex i.

Feasible exchanges

A set of vertex-disjoint cycles and chains.

- **Cycles** are formed by vertices from set P;
- **Chains** are initiated by an altruistic donor from set N followed by vertices from P.

Kidney Exchange Programs

Matching policies

- **Periodic** (Static)
  A matching run is performed periodically (e.g. each 3 months).

- **Greedy** (Dynamic)
  A matching run is performed whenever a new entity enters the pool.

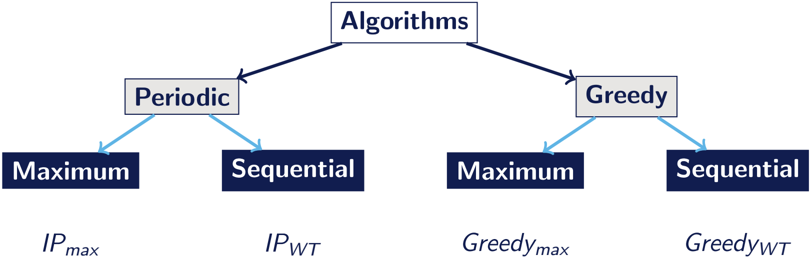
Kidney Exchange Programs

Objectives

- **Maximization**
  Choose solution with the largest number of transplants (randomly in case of multiple optimal).

- **Maximization of the number of pairs that waited longer (Sequential)**
  Preference is given to the pairs that has been in the pool for longer.

Algorithms



Algorithms
Periodic
Greedy
Maximum
Sequential
Maximum
Sequential
IP_{max}
IP_{WT}
$Greedy_{max}$
$Greedy_{WT}$

Integer Programming – IP_{max}

Cycle formulation

Let $\mathcal{C}(k)$ be the **set of all cycles** in G with **length at most** k. Define a variable x_c for each cycle $c \in \mathcal{C}(k)$:

$$x_c = \begin{cases} 1 & \text{if cycle } c \text{ is selected for the exchange,} \\ 0 & \text{otherwise.} \end{cases}$$

$$\text{Maximize} \quad \sum_{c \in \mathcal{C}(k)} |c| x_c \tag{1a}$$

$$\text{Subject to:} \quad \sum_{c : i \in c} x_c \leq 1 \qquad \forall i \in V \tag{1b}$$

$$x_c \in \{0, 1\} \qquad \forall c \in \mathcal{C}(k). \tag{1c}$$

- (1a): maximizes the number of transplants;
- (1b): every vertex is in at most one of the selected cycles.

Sequential Integer Programming – IP_{WT}

Let T be the **current matching period**.
For $t = T, T-1, \ldots, 1$ **sequentially solve** the following problems:

$$v_t^* = \max_x \sum_{c \in \mathcal{C}(k)} w_c^t x_c \tag{2a}$$

$$\text{Subject to: } \sum_{c: i \in c} x_c \leq 1 \qquad \forall i \in V \tag{2b}$$

$$\sum_{c \in \mathcal{C}} w_c^\tau x_c \geq v_\tau^* \qquad \tau = T, T-1, \ldots, t+1, t < T, \tag{2c}$$

$$x_c \in \{0, 1\} \qquad \forall c \in \mathcal{C}(k), \tag{2d}$$

where w_c^t is the **number of pairs in c that waited for t matching periods.**

- (2a): maximize the number of pairs that waited t matching periods;
- (2c): ensure the maximal number of transplants for pairs that waited longer.

Sequential Integer Programming – IP_{WT}

Let T be the **current matching period**.
For $t = T, T-1, \ldots, 1$ **sequentially solve** the following problems:

$$v_t^* = \max_x \sum_{c \in \mathcal{C}(k)} w_c^t x_c \tag{2a}$$

$$\text{Subject to: } \sum_{c: i \in c} x_c \leq 1 \qquad \forall i \in V \tag{2b}$$

$$\sum_{c \in \mathcal{C}} w_c^\tau x_c \geq v_\tau^* \qquad \tau = T, T-1, \ldots, t+1, t < T, \tag{2c}$$

$$x_c \in \{0, 1\} \qquad \forall c \in \mathcal{C}(k), \tag{2d}$$

where w_c^t is the **number of pairs in** c **that waited for** t **matching periods.**

- (2a): maximize the number of pairs that waited t matching periods;
- (2c): ensure the maximal number of transplants for pairs that waited longer.

Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool **Cycles** **Chains**

Greedy Algorithm – *Greedy*$_{max}$

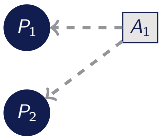
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool **Cycles** **Chains**

Greedy Algorithm – *Greedy$_{max}$*

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

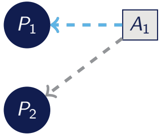
Pool **Cycles** **Chains**



Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

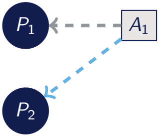
Pool **Cycles** **Chains**



Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

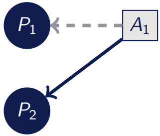
Pool **Cycles** **Chains**



Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool **Cycles** **Chains**

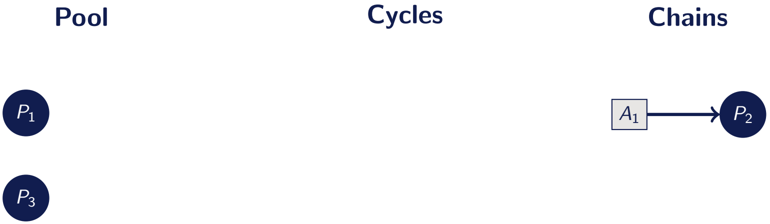


Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

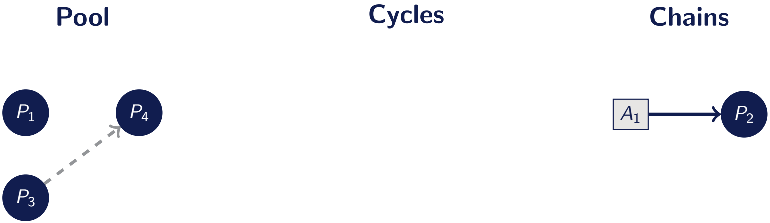
Greedy Algorithm – *Greedy*$_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



Greedy Algorithm – *Greedy$_{max}$*

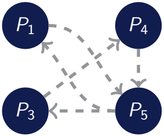
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool



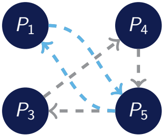
Cycles

Chains

Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool



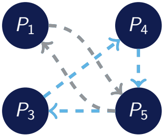
Cycles

Chains

Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool



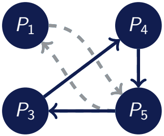
Cycles

Chains

Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool



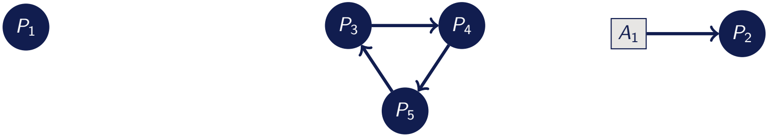
Cycles

Chains

Greedy Algorithm – *Greedy$_{max}$*

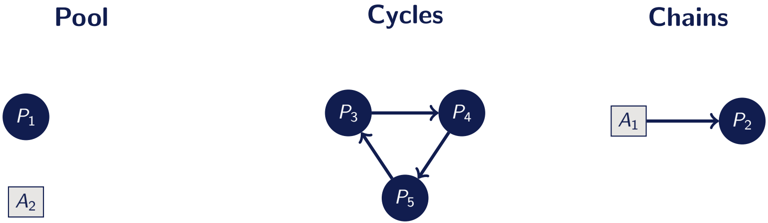
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).

Pool **Cycles** **Chains**



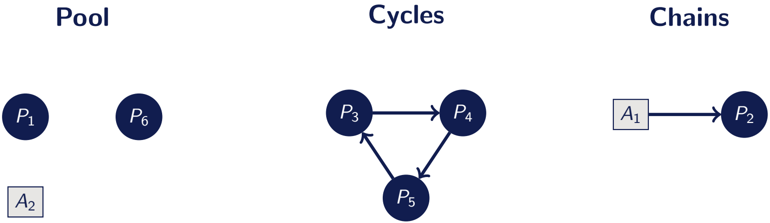
Greedy Algorithm – *Greedy$_{max}$*

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



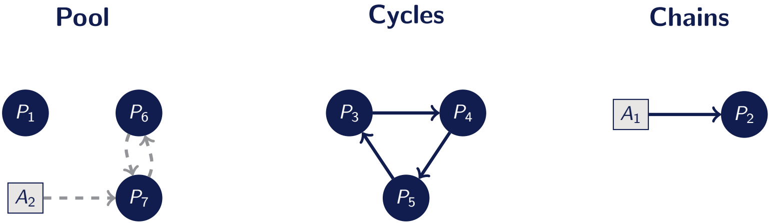
Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



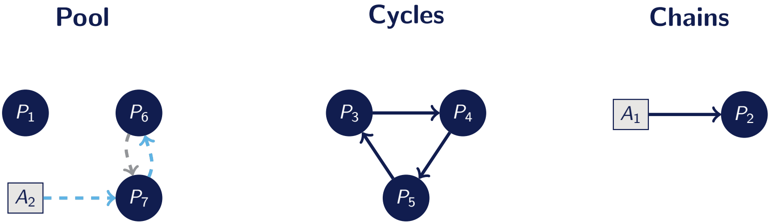
Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



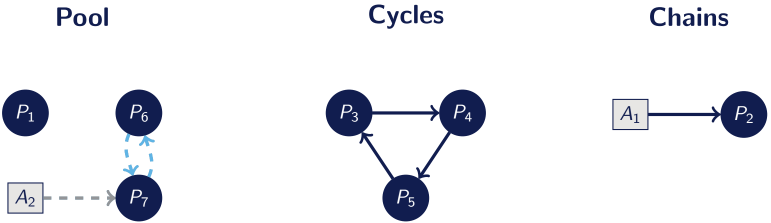
Greedy Algorithm – *$Greedy_{max}$*

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



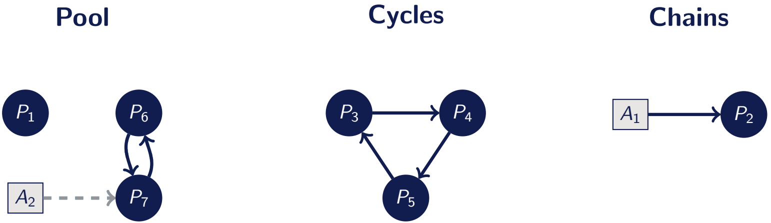
Greedy Algorithm – *Greedy*$_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



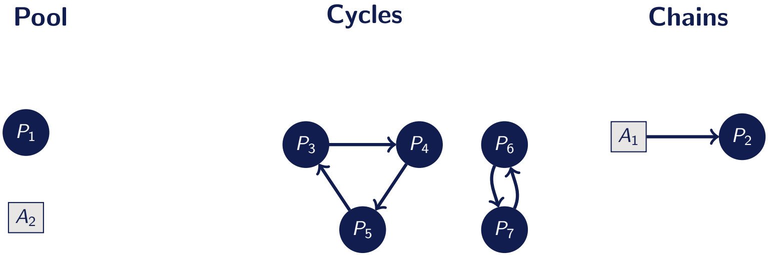
Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



Greedy Algorithm – $Greedy_{max}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, the one with **largest number of transplants** is selected (randomly in case of multiple optimal).



Greedy Sequential Algorithm – $Greedy_{WT}$

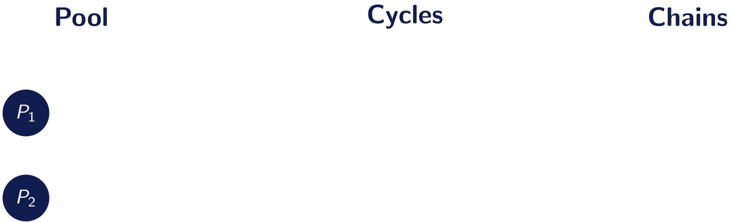
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Greedy Sequential Algorithm – $Greedy_{WT}$

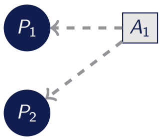
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool for longer**.

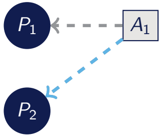
Pool **Cycles** **Chains**



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

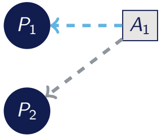
Pool **Cycles** **Chains**



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool for longer**.

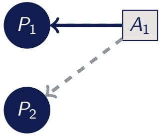
Pool **Cycles** **Chains**



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool **Cycles** **Chains**

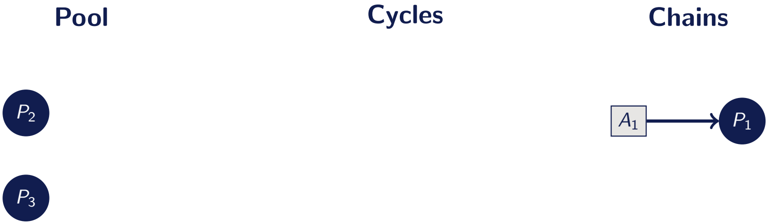


Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

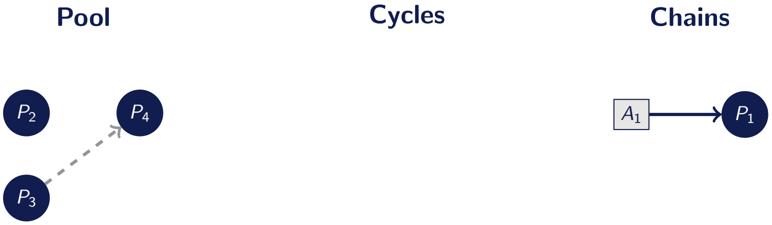
Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.



Greedy Sequential Algorithm – $Greedy_{WT}$

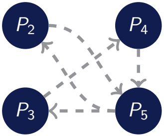
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



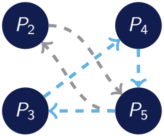
Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool for longer**.

Pool



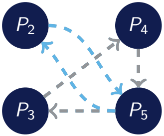
Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



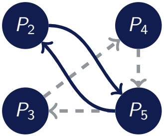
Cycles

Chains

Greedy Sequential Algorithm – *Greedy*$_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

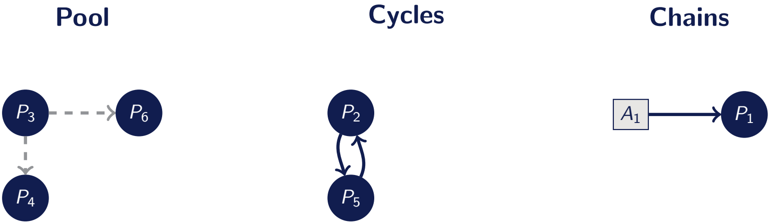
Pool

Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

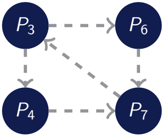
A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool for longer**.



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



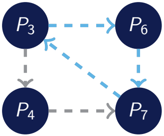
Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



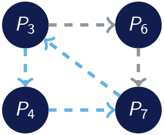
Cycles

Chains

Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool



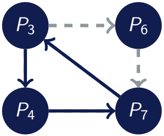
Cycles

Chains

Greedy Sequential Algorithm – *Greedy*$_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

Pool

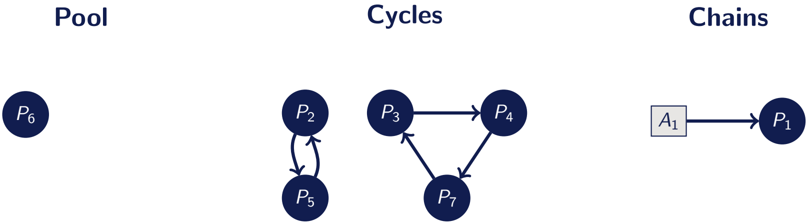


Cycles

Chains

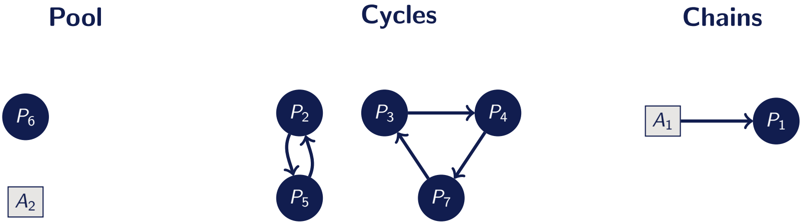
Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.



Greedy Sequential Algorithm – $Greedy_{WT}$

A matching algorithm **is run** whenever **a new incompatible pair or an altruistic donor joins the pool**. If there is more than one solution, preference is given to the one which **contains the pair that has been in the pool** for **longer**.

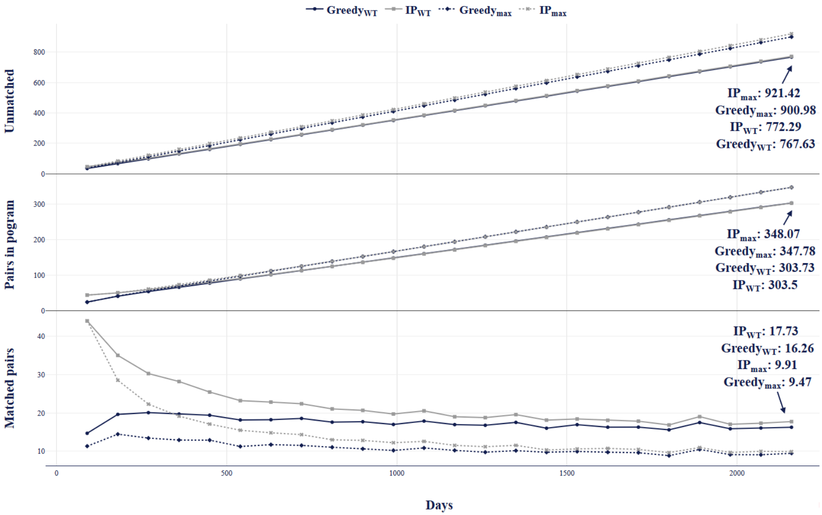


Computational analysis

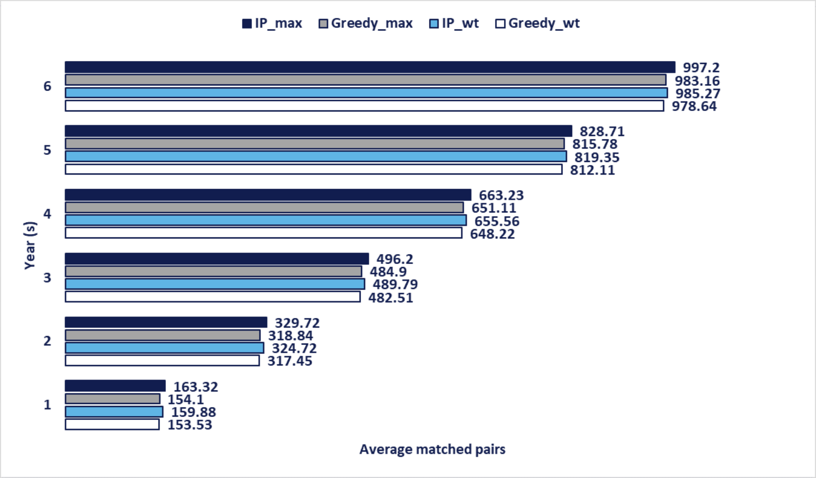
Computational tests

- **Machine:** Intel Core i7 CPU @ 2GHz, 8GB RAM
- Bound for cycles and chains length: $k = \mathbf{3}$
- The last donor in chains donate to the deceased donors waiting list
- Period: **90 days** (3 months)
- **100** instances
  - Generator presented in N. Santos, P. Tubertini, A. Viana and J. P. Pedroso: Kidney exchange simulation and optimization. *Journal of the Operational Research Society*, 68(12):1521-1532, 2017.
  - **6 years horizon**
  - **1652 pairs or altruistic donors** in average enter the pool for each instance (**only leave when matched**)
- To compare algorithms, the results obtained with the Greedy algorithms are accumulated at the end of each period.

Average waiting time



Number of matched pairs



Waiting Time vs Number of Transplants

	IP_{max}	IP_{WT}	$Greedy_{max}$	$Greedy_{WT}$
	348.07	**303.5**	347.78	*303.73*
	**997.2**	985.27	983.16	978.64

IP_{max} vs IP_{WT}:

 + 45

 + 11

Conclusions

Conclusions

- In this work we simulate different matching policies in order to identify how they affect pairs waiting time;
- For a six years horizon results show that average **waiting time of pairs in the program** are **reduced** (**45** days) for the formulation that **consider pairs waiting time**.
  - this is achieved by slightly **sacrificing** the total **number of transplants performed** (**11**) and **increasing average waiting times of matched pairs** (**8** days).

Acknowledgments

Thank you for your attention

This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme, and by National Funds through the Portuguese funding agency, FCT - Fundação para a Ciência e a Tecnologia, within project "mKEP - Models and optimisation algorithms for multicountry kidney exchange programs" (POCI-01-0145-FEDER-016677), by FCT project SFRH/BPD/101134/2014 and by COST Action CA15210, ENCKEP, supported by COST (European Cooperation in Science and Technology) – http://www.cost.eu/.