Quality and Fairness of Online Matching Algorithms for Kidney Exchange

Kelsey Lieberman & William Macke
with Li, Das and Ho

August 2, 2018

Background

- About 100, 000 people waiting for kidney transplants in the US (2016)

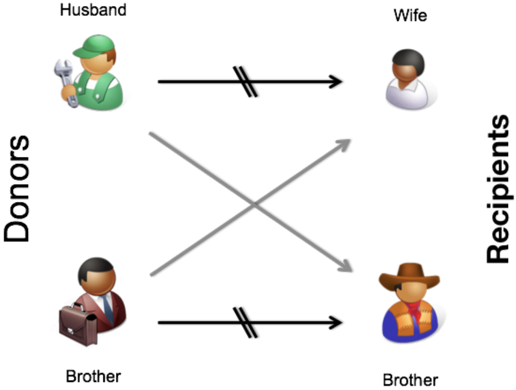
Background

- About 100, 000 people waiting for kidney transplants in the US (2016)
- In 2014, 17,107 kidney transplants took place, about 1/3 from living donors

Background

- About 100, 000 people waiting for kidney transplants in the US (2016)
- In 2014, 17,107 kidney transplants took place, about 1/3 from living donors
- Unfortunately, willing living donors are often not medically compatible.
- One option for them is to enter a kidney exchange program

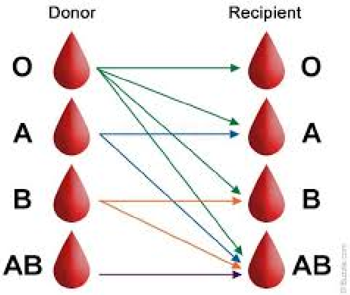
Incompatible Pairs



Compatibility

What makes a pair incompatible?

- Blood type
- Tissue type (measured by PRA)



Quality of Match

LKDPI:

- Blood type of donor
- Blood type of recipient
- Age of donor
- Sex of both
- Weight/BMI of both
- Cigarette use of donor
- If donor is African American
- PRA of recipient

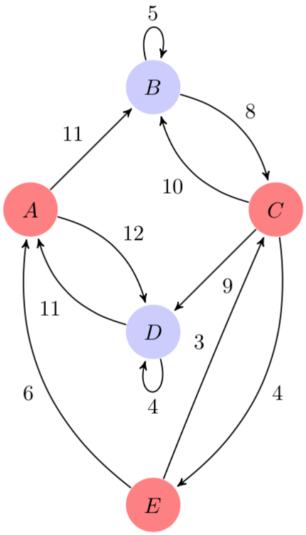
Quality of Match

LKDPI:

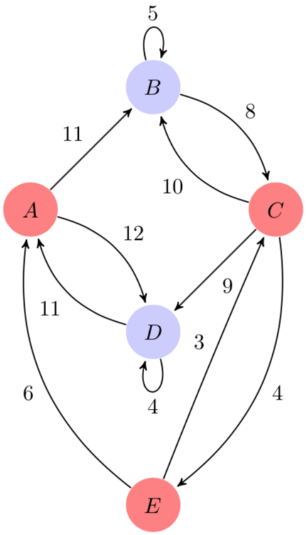
- Blood type of donor
- Blood type of recipient
- Age of donor
- Sex of both
- Weight/BMI of both
- Cigarette use of donor
- If donor is African American
- PRA of recipient

LKDPI $\rightarrow$ EGS (Expected Graft Survival)

Formulation

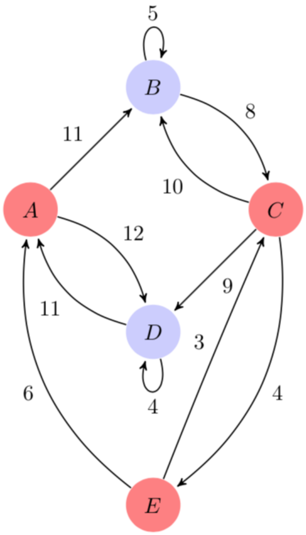


Formulation



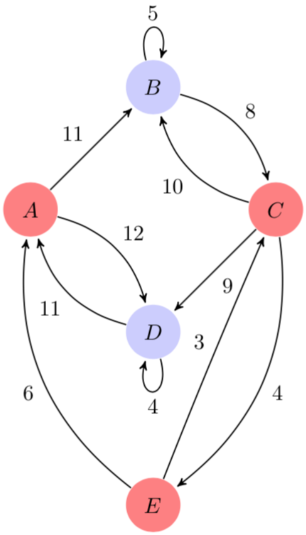
- Start with K incompatible pairs

Formulation



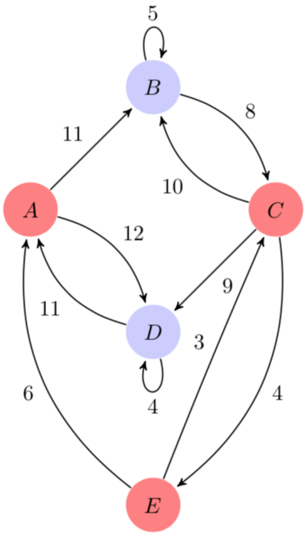
- Start with K incompatible pairs
- 1 compatible pair arrives at each of T time steps

Formulation



- Start with K incompatible pairs
- 1 compatible pair arrives at each of T time steps
- Compatible pairs only participate if they are matched **immediately** and the recipient receives a **higher** quality match

Formulation



- Start with K incompatible pairs
- 1 compatible pair arrives at each of T time steps
- Compatible pairs only participate if they are matched **immediately** and the recipient receives a **higher** quality match
- Cycles can include up to 3 pairs

Problem Statement

- How can we optimize matches within a dynamic population without knowing the future of pair arrivals?

Problem Statement

- How can we optimize matches within a dynamic population without knowing the future of pair arrivals?
- Is this method fair?

Optimization technique

- Start by assuming we know the future

Optimization technique

- Start by assuming we know the future
- Use a Linear Program to optimize

Optimization technique

- Start by assuming we know the future
- Use a Linear Program to optimize
  - $0 \leq x_{t,i} \leq 1$ - Match between pairs t and i

Optimization technique

- Start by assuming we know the future
- Use a Linear Program to optimize
    - o $0 \leq x_{t,i} \leq 1$ - Match between pairs t and i
- Use an Integer Program to optimize

Optimization technique

- Start by assuming we know the future
- Use a Linear Program to optimize
  - o $0 \leq x_{t,i} \leq 1$ - Match between pairs t and i
- Use an Integer Program to optimize
  - o $x_{t,i} = 0$ or 1

Optimization technique

- Start by assuming we know the future
- Use a Linear Program to optimize
  - o $0 \leq x_{t,i} \leq 1$ - Match between pairs t and i
- Use an Integer Program to optimize
  - o $x_{t,i} = 0$ or 1
  - o Can perform optimization using IP solvers(Gurobi, CBC)

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T\}$

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T\}$
- $i \in I = \{1, 2, 3 ... K\}$

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T, T + 1 ... T + K\}$
- $i \in I = \{1, 2, 3 ... K\}$

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 \ldots T, T + 1 \ldots T + K\}$
- $i \in I = \{0, 1, 2 \ldots K\}$

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T, T + 1 ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $x_{t,0}$ - Compatible Pair matching with self

Optimizing for Oracle: Variables

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T, T + 1 ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $x_{t,0}$ - Compatible Pair matching with self
- $w_{t,i}$ - Weight of match between pairs t and i

Optimizing for Oracle: Primal IP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 ... T, T + 1 ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $x_{t,0}$ - Compatible Pair matching with self
- $w_{t,i}$ - Weight of match between pairs t and i

$$\text{maximize} \sum_{t,i} w_{t,i} x_{t,i}$$

subject to

Optimizing for Oracle: Primal IP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 \dots T, T + 1 \dots T + K\}$
- $i \in I = \{0, 1, 2 \dots K\}$
- $x_{t,0}$ - Compatible Pair matching with self
- $w_{t,i}$ - Weight of match between pairs t and i

$$
\begin{aligned}
&\text{maximize} \sum_{t,i} w_{t,i} x_{t,i} \\
&\text{subject to} \\
&\sum_{i} x_{t,i} \leq 1, && \forall t \leq T
\end{aligned}
$$

Optimizing for Oracle: Primal IP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $x_{t,i}$ - Match between pairs t and i
- $t \in C = \{1, 2, 3 \ldots T, T + 1 \ldots T + K\}$
- $i \in I = \{0, 1, 2 \ldots K\}$
- $x_{t,0}$ - Compatible Pair matching with self
- $w_{t,i}$ - Weight of match between pairs t and i

$$
\begin{aligned}
&\text{maximize} \sum_{t,i} w_{t,i} x_{t,i} \\
&\text{subject to} \\
&\sum_{i} x_{t,i} \le 1, && \forall t \le T \\
&\sum_{t} x_{t,i} + \sum_{j \in I} x_{(i+T),j} \le 1, && \forall i \ge 1
\end{aligned}
$$

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.
- Each Constraint becomes a variable and each variable becomes a constraint.

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.
- Each Constraint becomes a variable and each variable becomes a constraint.
- $\sum_i x_{t,i} \leq 1 \quad \forall t \leq T$ - α_t

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.
- Each Constraint becomes a variable and each variable becomes a constraint.
- $\sum_i x_{t,i} \leq 1 \quad \forall t \leq T$ - α_t
- $\sum_t x_{t,i} + \sum_{j \in I} x_{(i+T),j} \leq 1 \quad \forall i \geq 1$ - β_i

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.
- Each Constraint becomes a variable and each variable becomes a constraint.
- $\sum_i x_{t,i} \leq 1 \quad \forall t \leq T$ - α_t
- $\sum_t x_{t,i} + \sum_{j \in I} x_{(i+T),j} \leq 1 \quad \forall i \geq 1$ - β_i
- α_t - Rough measure of value of compatible pair

Optimizing for Oracle: Finding the Dual

- By relaxing IP above to LP, we can solve for dual problem.
- Each Constraint becomes a variable and each variable becomes a constraint.
- $\sum_i x_{t,i} \leq 1 \quad \forall t \leq T$ - α_t
- $\sum_t x_{t,i} + \sum_{j \in I} x_{(i+T),j} \leq 1 \quad \forall i \geq 1$ - β_i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

7

Optimizing for Oracle: Dual LP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $t \in C = \{1, 2, 3...T, T+1, ...T+K\}$
- $i \in I = \{0, 1, 2...K\}$
- $w_{t,i}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

Optimizing for Oracle: Dual LP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $w_{t,i}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$\text{minimize} \sum_t \alpha_t + \sum_i \beta_i$$

subject to

Optimizing for Oracle: Dual LP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $w_{t,i}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$\begin{aligned}
&\text{minimize} \sum_t \alpha_t + \sum_i \beta_i \\
&\text{subject to} \\
&w_{t,i} - \alpha_t - \beta_i \leq 0, \qquad \forall t \leq T, i
\end{aligned}$$

Optimizing for Oracle: Dual LP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $w_{t,i}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$
\begin{aligned}
&\text{minimize} \sum_t \alpha_t + \sum_i \beta_i \\
&\text{subject to} \\
&w_{t,i} - \alpha_t - \beta_i \leq 0, \qquad \forall t \leq T, i \\
&w_{t,i} - \beta_{t-T} - \beta_i \leq 0, \qquad \forall t > T, i
\end{aligned}
$$

Optimizing for Oracle: Dual LP

- T - Number of Compatible Pairs
- K - Number of Incompatible Pairs
- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $w_{t,i}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$\begin{aligned}
&\text{minimize} \sum_t \alpha_t + \sum_i \beta_i \\
&\text{subject to} \\
&w_{t,i} - \alpha_t - \beta_i \le 0, \qquad \forall t \le T, i \\
&w_{t,i} - \beta_{t-T} - \beta_i \le 0, \qquad \forall t > T, i
\end{aligned}$$

$$\text{Choose } i = \arg\max_{i'} w_{t,i'} - \beta_{i'}$$

Algorithm

- Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.
  - o Predict β based on two sets of data.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.
    - o Predict β based on two sets of data.
        1. Demographics information of pairs.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.
  - o Predict β based on two sets of data.
    1. Demographics information of pairs.
    2. β predicted by LP on incompatible pool only.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.
  - o Predict β based on two sets of data.
    1. Demographics information of pairs.
    2. β predicted by LP on incompatible pool only.
  - o Performs assignment based on rule given by dual above.

Algorithm

I Greedy: Attempts to match each arriving compatible pair to maximize that cycle weight.

II Online Dual Assignment using Shadow Survival Estimates (ODASSE):

- Attempts to predict future based on given information and past experiences.
    - o Predict β based on two sets of data.
        1. Demographics information of pairs.
        2. β predicted by LP on incompatible pool only.
    - o Performs assignment based on rule given by dual above.

III Oracle: Upper bound, we assume we know the future and perform the optimal matching using an integer program.

Experimental Protocol

- Simulations based on demographics information of pairs generated by parameters from Barnes Jewish Hospital.

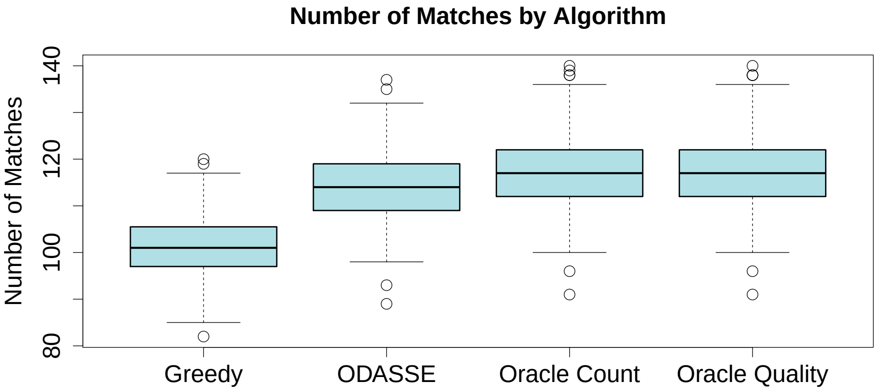
Experimental Protocol

- Simulations based on demographics information of pairs generated by parameters from Barnes Jewish Hospital.
- All experiments shown consisted of 500 simulations of 50 arriving compatible pairs with a pool size of 100 incompatible pairs.

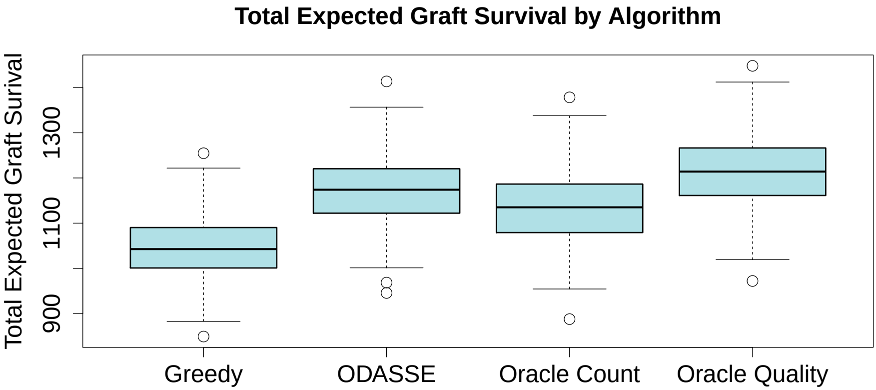
Experimental Protocol

- Simulations based on demographics information of pairs generated by parameters from Barnes Jewish Hospital.
- All experiments shown consisted of 500 simulations of 50 arriving compatible pairs with a pool size of 100 incompatible pairs.
- Training data is drawn from past simulations.

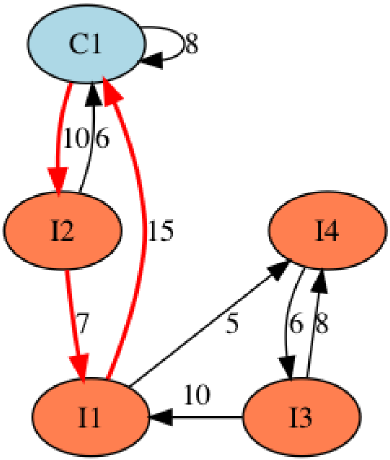
Preliminary Results



Preliminary Results



Three Cycles



Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3...T, T+1, ...T+K\}$
- $i \in I = \{0, 1, 2...K\}$
- $j \in I = \{0, 1, 2...K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self
- $x_{t,i,0}$ - 2 Cycle Match

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self
- $x_{t,i,0}$ - 2 Cycle Match
- $w_{t,i,j}$ - Weight of cycle of pairs t,i and j

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 \ldots T, T+1, \ldots T+K\}$
- $i \in I = \{0, 1, 2 \ldots K\}$
- $j \in I = \{0, 1, 2 \ldots K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self
- $x_{t,i,0}$ - 2 Cycle Match
- $w_{t,i,j}$ - Weight of cycle of pairs t,i and j

$$\text{maximize} \sum_{t,i,j} w_{t,i,j} x_{t,i,j}$$

subject to

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self
- $x_{t,i,0}$ - 2 Cycle Match
- $w_{t,i,j}$ - Weight of cycle of pairs t,i and j

$$
\begin{aligned}
&\text{maximize} \sum_{t,i,j} w_{t,i,j} x_{t,i,j} \\
&\text{subject to} \\
&\sum_{i,j} x_{t,i,j} \leq 1, && \forall t \leq T
\end{aligned}
$$

Three Cycle: Primal IP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $x_{t,i,j}$ - Cycle of pairs t,i and j
- $x_{t,0,0}$ - Compatible Matching with self
- $x_{t,i,0}$ - 2 Cycle Match
- $w_{t,i,j}$ - Weight of cycle of pairs t,i and j

$$
\begin{aligned}
&\text{maximize} \sum_{t,i,j} w_{t,i,j} x_{t,i,j} \\
&\text{subject to} \\
&\sum_{i,j} x_{t,i,j} \leq 1, && \forall t \leq T \\
&\sum_{t,j} x_{t,i,j} + \sum_{t,j} x_{t,j,i} + \sum_{j,j' \in I} x_{(i+T),j,j'} \leq 1, && \forall i \geq 1
\end{aligned}
$$

Three Cycle: Dual LP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $w_{t,i,j}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$\text{minimize} \sum_t \alpha_t + \sum_i \beta_i$$

$$\text{subject to}$$

Three Cycle: Dual LP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $w_{t,i,j}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$\text{minimize} \sum_t \alpha_t + \sum_i \beta_i$$

subject to

$$w_{t,i,j} - \alpha_t - \beta_i - \beta_j \leq 0, \qquad \forall t \leq T, i, j$$

Three Cycle: Dual LP

- $t \in C = \{1, 2, 3 ... T, T + 1, ... T + K\}$
- $i \in I = \{0, 1, 2 ... K\}$
- $j \in I = \{0, 1, 2 ... K\}$
- $w_{t,i,j}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

$$
\begin{aligned}
&\text{minimize} \sum_t \alpha_t + \sum_i \beta_i \\
&\text{subject to} \\
&w_{t,i,j} - \alpha_t - \beta_i - \beta_j \leq 0, \qquad \forall t \leq T, i, j \\
&w_{t,i,j} - \beta_{t-T} - \beta_i - \beta_j \leq 0, \quad \forall t > T, i, j
\end{aligned}
$$

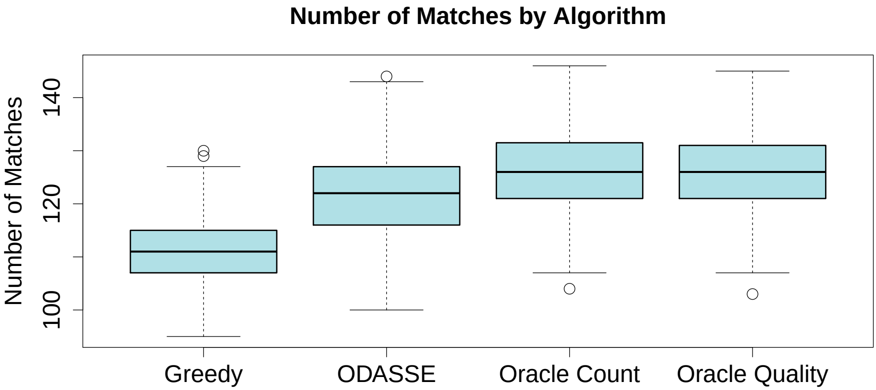
Three Cycle: Dual LP

- $t \in C = \{1, 2, 3 \dots T, T+1, \dots T+K\}$
- $i \in I = \{0, 1, 2 \dots K\}$
- $j \in I = \{0, 1, 2 \dots K\}$
- $w_{t,i,j}$ - Weight of match between pairs t and i
- α_t - Rough measure of value of compatible pair
- β_i - Rough measure of value of incompatible pair

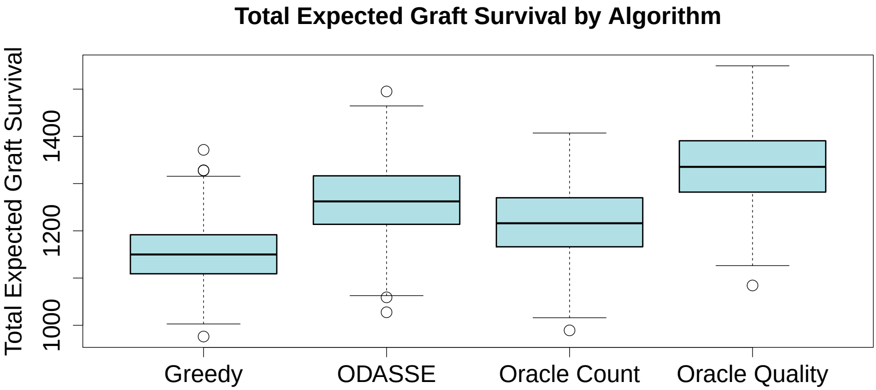
$$
\begin{aligned}
&\text{minimize} \sum_t \alpha_t + \sum_i \beta_i \\
&\text{subject to} \\
&w_{t,i,j} - \alpha_t - \beta_i - \beta_j \le 0, \qquad \forall t \le T, i, j \\
&w_{t,i,j} - \beta_{t-T} - \beta_i - \beta_j \le 0, \quad \forall t > T, i, j
\end{aligned}
$$

$$
\text{Choose } i, j = \arg\max_{i', j'} w_{t,i',j'} - \beta_{i'} - \beta_{j'}
$$

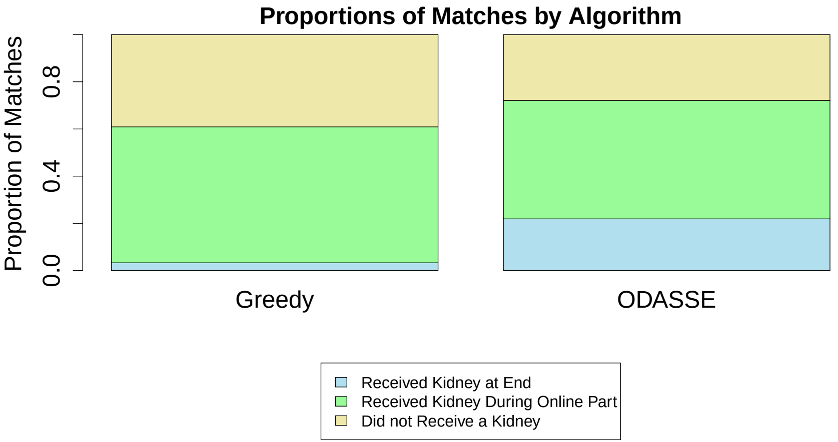
Number of Matches



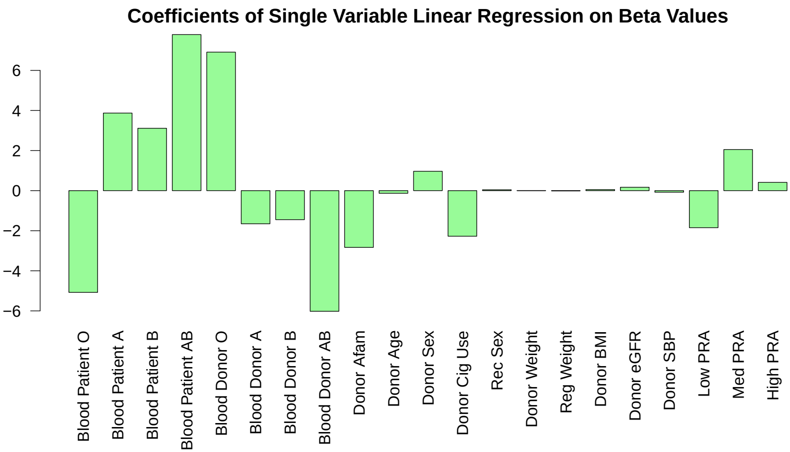
Quality of Matches



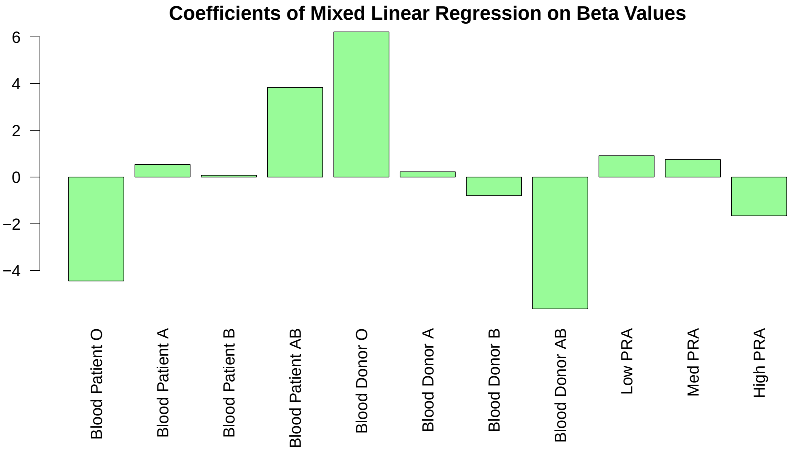
Time of Matches



Interpreting Beta



Interpreting Beta



Analyzing Fairness

Do the algorithms result in differential impacts for specific groups?

Groups to consider:

- Demographic groups
- Groups based on beta values

Analyzing Fairness

Do the algorithms result in differential impacts for specific groups?

Groups to consider:

- Demographic groups
- Groups based on beta values

Outcomes to consider:

- **Proportion** of pairs from that get matched
- **Quality** a pair receives if matched
- **Time** a pair waits to receive a match

Fairness by β

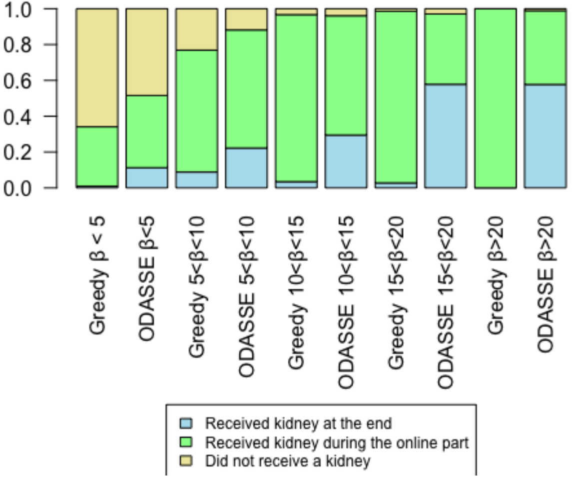


Figure: Incompatible pairs with $\beta < 5$ receive 45.5% more matches with ODASSE algorithm than greedy

Fairness

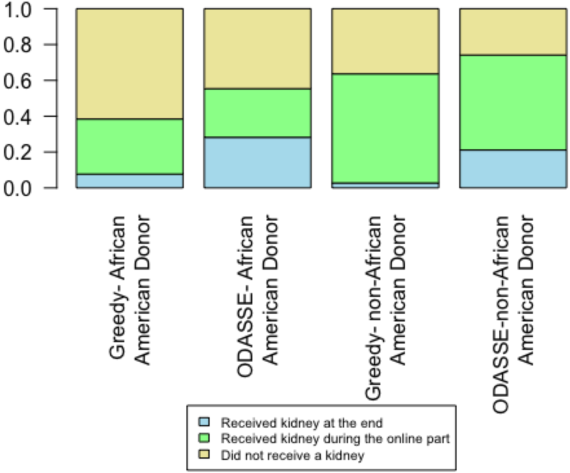


Figure: Incompatible pairs with African American donors receive 44.3% more matches with ODASSE algorithm than greedy

Fairness in Blood Type and Tissue Type

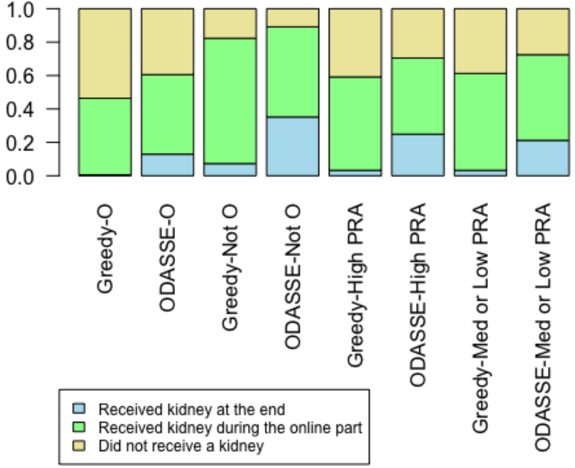


Figure: Incompatible pairs with recipients with type O blood receive 30.6% more matches with ODASSE algorithm than greedy

Received Quality Over Time

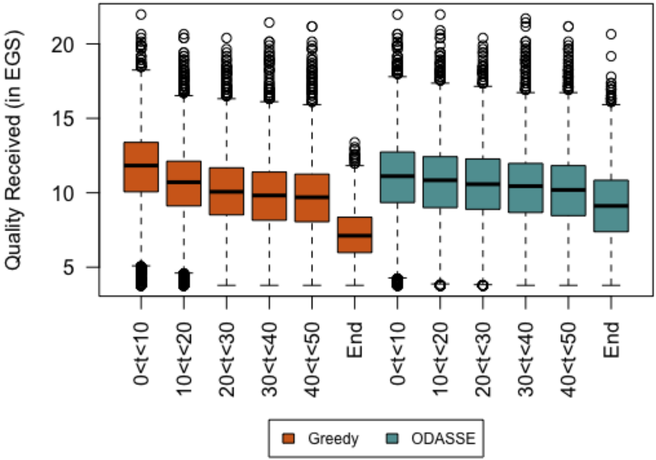


Figure: Quality Recipient Receives by Time Matched

Summary

- ODASSE improves the social welfare

Summary

- ODASSE improves the social welfare
- ODASSE improves number of transplants performed

Summary

- ODASSE improves the social welfare
- ODASSE improves number of transplants performed
- ODASSE improves fairness among hard-to-match pairs

Summary

- ODASSE improves the social welfare
- ODASSE improves number of transplants performed
- ODASSE improves fairness among hard-to-match pairs
- Improved outcomes do not come at the cost of reduced fairness

Future work

- Expanding the formulation to include arrival and departure protocols for incompatible pairs
- Adding predictions of future *compatible* pairs to improve algorithm