Maximum-expectation matching under recourse

João Pedro Pedroso

Based on "Maximum-expectation matching under recourse",
joint work with Shiro Ikeda

Madrid 28-29 November 2016
CYTED Workshop

Kidney exchange programs

- Motivation
- Models
- Reconfigurations

Information systems for health care

- Expected to be one of the areas where more resources will be applied in the next few years
- Has issues involving the many disciplines, including operations research, computer science, informatics, . . .
- Information systems have a huge impact in terms of
  - economy
  - social benefits
  - work rationalization
  - reliability

Kidney Failure Treatments

- Kidney failure
  - One kidney ⟶ OK
  - Both kidneys ⟶ Dialysis or Transplantation
- Dialysis vs Transplantation
  - Transplantation yields longer survivability
  - Transplantation yields a better quality of life
  - Dialysis is more expensive than transplantation; values for Portugal:
    - Hemodialysis ⟶ 30K euro per year per person
    - Transplantation: 30K euro once + 10K euro year

Kidney Failure Treatments

I don't care what day it is.
Four hours is four hours.

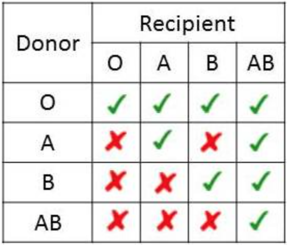
- Objective: ⟶ carry out the *maximum* possible number of (successful) transplants

Sources of kidneys for transplantation

- Deceased donors
  - very large waiting lists (5 years or more waiting)
- Living donors:
  - relatives, spouse, friends, altruistic donors
  - many ethical and legal issues (varies with country)
    - e.g. no commercial transaction of kidneys is generally accepted

Sources of incompatibility

- Blood type compatibilities



Donor	Recipient			
	O	A	B	AB
O	✓	✓	✓	✓
A	✗	✓	✗	✓
B	✗	✗	✓	✓
AB	✗	✗	✗	✓

- Tissue type incompatibility
  - HLA (Human Leukocyte Antigens)
  - . . .

Background: kidney exchange programs

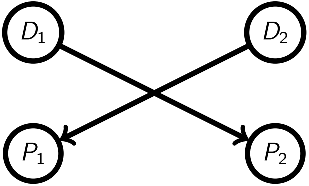
- in many countries, recent legislation allows patients needing a kidney transplant to receive it from a living donor
- what to do when the transplant from that donor is not possible?
    - blood type
    - other incompatibilities
- patient-donor pair may enter a kidney exchange program (KEP)

Kidney exchange programs

- KEPs were first proposed by (Rapaport, 1986)
- First transplants within a KEP were done in South Korea, 1991
- Many countries have now KEPs (USA, Switzerland, Turkey, Romania, Netherlands, UK, Canada, Australia, New Zealand, Spain)
- A KEP started in Portugal in 2011; presently, transplants are routinely performed

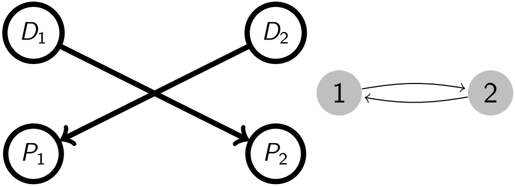
Kidney exchanges

- Suppose there are two patient-donor pairs (D_1, P_1) and $(D_2, P2)$
- Donor D_1 is willing to give kidney to patient P_1 but they are incompatible
- The same for pair D_2, P_2
- D_1 is compatible with P_2 and D_2 is compatible with P_1
- Then, D_1 can give a kidney to P_2 and D_2 can give a kidney to P_1



Kidney 2-exchanges

- allow two patients in incompatible pairs to exchange their donors
- each patient receives a compatible kidney from the donor of the other pair



Incompatible pairs $P_1 - D_1$ and $P_2 - D_2$ exchange donors

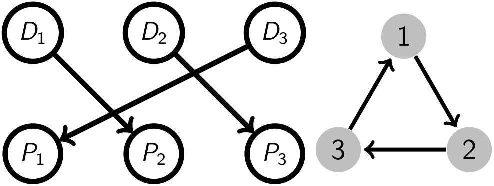
- P_1 receives a transplant from D_2 and vice versa

Graph representation:

- vertices are patient-donor pairs
- arcs link a donor to compatible patients

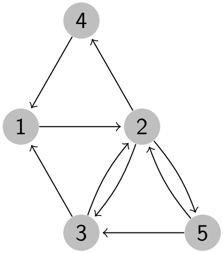
Kidney 3-exchanges

- The idea can be easily extended to 3 or more pairs:



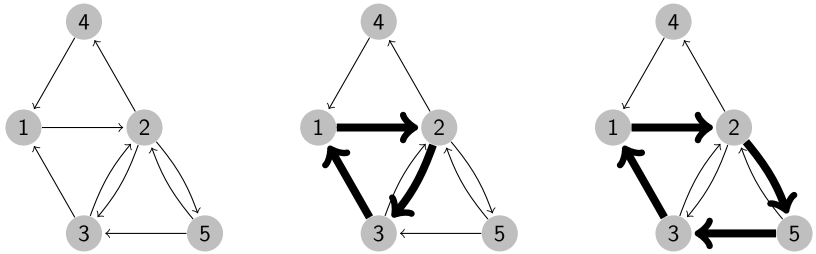
- Representation with a directed exchange graph:
  - each incompatible pair (D_i, P_i) corresponds to a node i
  - there exists an arc between i and j if donor D_i can give a kidney to patient P_j
  - a cycle with k nodes in this graph corresponds to a k-exchange

Kidney exchanges: example



- instance with five pairs
- what is the maximum number of transplants?
- what if the allowed number of simultaneous transplants is limited?

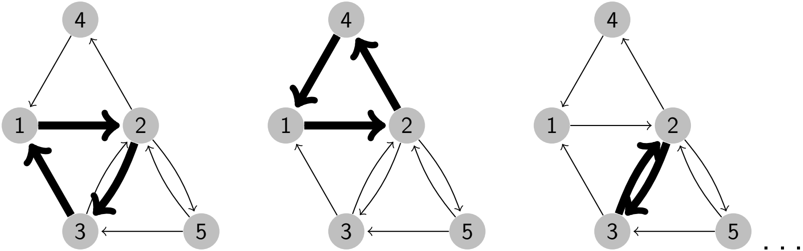
Kidney exchanges: example



- feasible exchange: a set of vertex-disjoint cycles (*e.g.*, 1 – 2 – 3 – 1)
- size of an exchange: sum of the lengths of its cycles
- maximum exchange in this example: 4 (cycle 1 – 2 – 5 – 3 – 1)

Kidney exchanges: maximum cycle size

- In many situations the length of each cycle is limited
- If maximum cycle size is $K = 3$, several solutions are possible.



Kidney exchanges: why limiting size

- Two main reasons:
  - usually, all transplants in a cycle should be done at same time
    - someone could withdraw from the program
  - last-minute incompatibility test (crossmatch, just before transplantation)
    - if positive, no transplantation can be done for any pair in this cycle
    - (rearrangements may change the previous limitation)
- However, optimum number of transplants increases with maximum size allowed
- Most programs have $k = 2$ or $k = 3$

Kidney Exchange Model

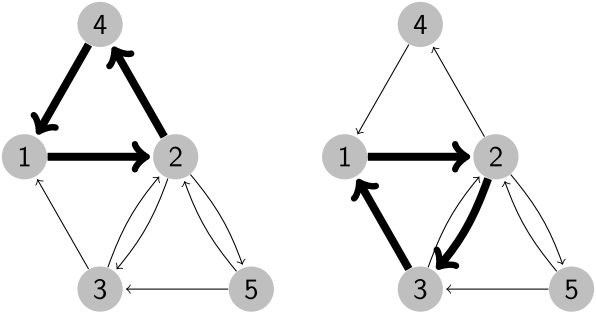
- Given:
  - a pool of n incompatible donor-patient pairs
  - the compatibility between all donors and all patients
- find the maximum number of kidney exchanges with cycles of size at most~k

Complexity

- Is this problem easy to solve?
  - YES, if $k = 2$ or no limit is imposed on the size of the cycles
  - NO, if $k = 3, 4, 5, \ldots$
- If $k = 2$ the problem reduces to finding a maximum matching in a undirected graph, which can be solved efficiently (Edmonds 1965)
- If no limit is imposed on the size of the cycles the problem can be formulated as an assignment problem (can be solved efficiently by hungarian algorithm)
- The problem is NP-hard for $k = 3, 4, 5, \ldots$ (hence, no polynomial algorithms are known to solve it)

Mathematical programming formulations

- There are several possibilities for modeling the problem in mathematical programming
- One of the most successful is the cycle formulation:
  - enumerate all cycles in the graph with length at most K
  - for each cycle c, let variable x_c be 1 if c is chosen, 0 otherwise
  - every feasible solution corresponds to a set of vertex-disjoint cycles



Cycle formulation

$$
\begin{aligned}
\text{maximize} \quad & \sum_{c} w_c x_c && (1)\\
\text{subject to} \quad & \sum_{c:i \in c} x_c \le 1 \quad \forall i && (2)\\
& x_c \in \{0,1\} \quad \forall c
\end{aligned}
$$

- case of $0-1$ weights: $w_c = |c|$, (length of cycle˜c)
- objective: maximize the weight of the exchange
- constraints: every vertex is at most in one cycle (*i.e.*, donate/receive at most one kidney)
- difficulty: number of variables

Reconfigurations

Maximizing expectation

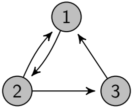
- How to optimize if there is some probability of vertex/arc failure?
  - vertex failure: due to some patient/donor become ill, or otherwise unavailable
  - arc failure:
    - a last-minute incompatibility test (crossmatch) is performed just before the transplantation
    - if any is positive, no transplantation involving this arc is possible

Maximizing expectation: model

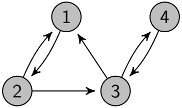
- Basis: cycle formulation
- Standard approach: cycle's value is its number of arcs (*i.e.*, the *number of transplants*)
- Our proposal: use the expectation of the number of transplants instead
- Problem: not straightforward to tackle...
  1. computation of the expectation is heavy, even for small cycles
  2. optimization is just a small part in the solution process...

Maximizing expectation: weighting cycles

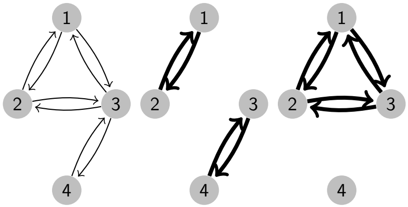
- No recourse: give a weight to each cycle based on its reliability, but no rearrangements of the matching are allowed
- Internal recourse: rearrangements are possible, as long as they involve only vertices of a cycle



- Subset recourse: rearrangements are possible, as long as they involve only a cycle extended with small subset of vertices



Internal recourse: Unreliable vertices



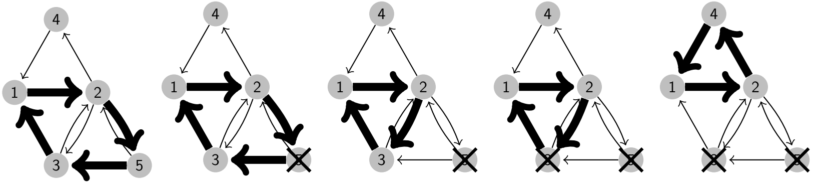
Solution procedure: implementation

- Implementation
  - contact selected pairs
  - verify solution (check back outs)
  - make last-minute compatibility check
  - make transplants

More on reconfigurations

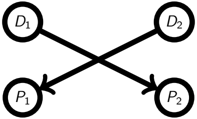
More on reconfigurations:

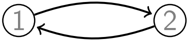
- In the previous cases, we allowed for ONE reconfiguration
- What if we allow *more than one*?
  - there is no natural limit on this number
  - *e.g.*, if two cycles fail, why not reassign the remaining pairs?

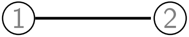


Reconfigurations:

- Caveat:
  - we will not be treating the general case
- Simplification:
  - considering only cycles of length 2
  - graph: undirected, edge when two patients can exchange donors

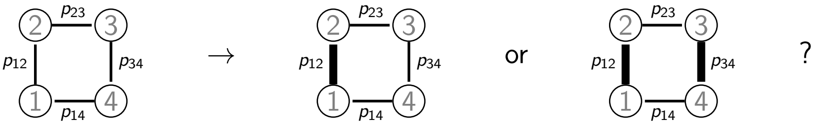






The story

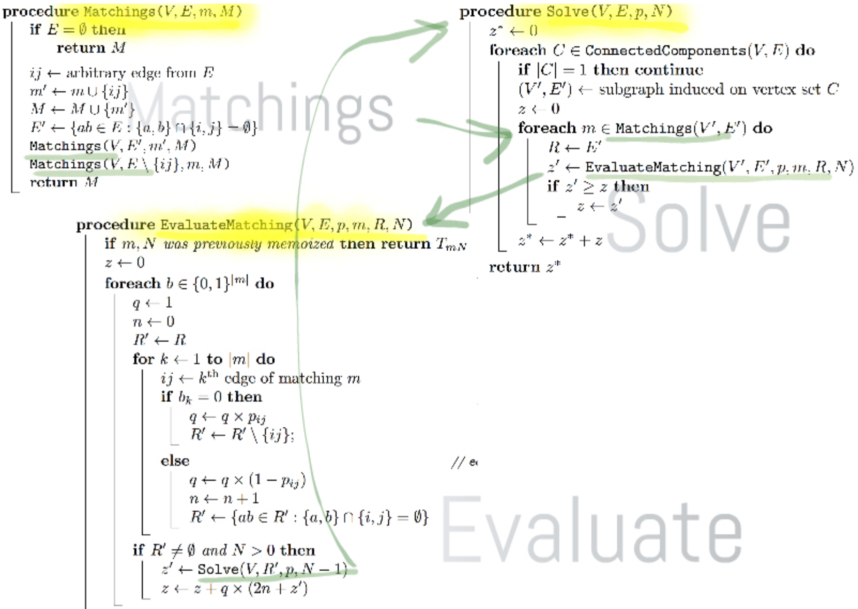
- case: limit to $k = 2 \rightarrow$ polynomial
- first approach:
  - enumerate all maximal-matchings
  - choose the one with best expectation
- but. . . maximum-expectation matching may be non-maximal



Some properties:

1. maximum-expectation matching may be non-maximal
2. with no limit on the number of observations, there is maximum-expectation matching with one edge per observation
3. as a consequence: maximum-expectation matching is not in EXPSPACE. . .
   - . . . any hope?

Algorithm

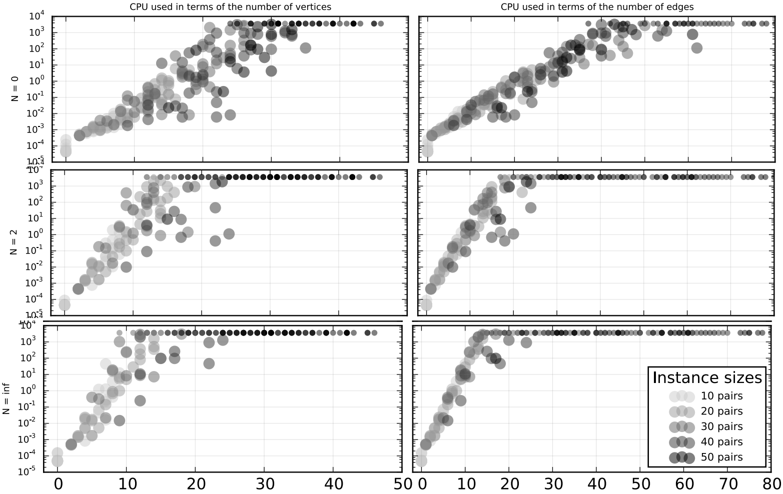


```
procedure Matchings($V, E, m, M$)
    if $E = \emptyset$ then
        return $M$
    $ij \leftarrow$ arbitrary edge from $E$
    $m' \leftarrow m \cup \{ij\}$
    $M \leftarrow M \cup \{m'\}$
    $E' \leftarrow \{ab \in E : \{a, b\} \cap \{i, j\} = \emptyset\}$
    Matchings($V, E', m', M$)
    Matchings($V, E \setminus \{ij\}, m, M$)
    return $M$
```

```
procedure Solve($V, E, p, N$)
    $z^* \leftarrow 0$
    foreach $C \in$ ConnectedComponents($V, E$) do
        if $|C| = 1$ then continue
        $(V', E') \leftarrow$ subgraph induced on vertex set $C$
        $z \leftarrow 0$
        foreach $m \in$ Matchings($V', E'$) do
            $R \leftarrow E'$
            $z' \leftarrow$ EvaluateMatching($V', E', p, m, R, N$)
            if $z' \geq z$ then
                $z \leftarrow z'$
        $z^* \leftarrow z^* + z$
    return $z^*$
```

```
procedure EvaluateMatching($V, E, p, m, R, N$)
    if $m, N$ was previously memoized then return $T_{mN}$
    $z \leftarrow 0$
    foreach $b \in \{0, 1\}^{|m|}$ do
        $q \leftarrow 1$
        $n \leftarrow 0$
        $R' \leftarrow R$
        for $k \leftarrow 1$ to $|m|$ do
            $ij \leftarrow k^{\text{th}}$ edge of matching $m$
            if $b_k = 0$ then
                $q \leftarrow q \times p_{ij}$
                $R' \leftarrow R' \setminus \{ij\}$;
            else                                             // e
                $q \leftarrow q \times (1 - p_{ij})$
                $n \leftarrow n + 1$
                $R' \leftarrow \{ab \in R' : \{a, b\} \cap \{i, j\} = \emptyset\}$
        if $R' \neq \emptyset$ and $N > 0$ then
            $z' \leftarrow$ Solve($V, R', p, N - 1$)
            $z \leftarrow z + q \times (2n + z')$
```

Behavior



CPU used in terms of the number of vertices
CPU used in terms of the number of edges
N = 0
N = 2
N = inf
10^{4}
10^{3}
10^{2}
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
0
10
20
30
40
50
0
10
20
30
40
50
60
70
80
Instance sizes
10 pairs
20 pairs
30 pairs
40 pairs
50 pairs

Limited recourse

- often there is a limit in the allowed number of observations/reconfigurations
- N-recourse: matching such that solution must be reached within N *observations*
  - $N = 0 \rightarrow$ standard matching
  - $N = \infty \rightarrow$ unlimited case
- difficulty:
  - solvable in polynomial time for $N = 0$
  - complexity increases with N
  - ∞-recourse intractable

Practical approach:

- Initial solution for $N = 0$
- Increment N until
  - additional gain acceptably low, or
  - computational time excessive

Solution

- Under limited recourse → no longer a binary tree
- On each node/observation one may optimally propose multiple edges
- Children of the node:
  - must include all the patterns of success or failure edges proposed
- Example: at a given observation:
  - matching: pairs {A,B} and {F,G}
  - if {A,B} and {F,G} succeed:
    - matching: {H,I} . . .
  - if {A,B} succeeds and {F,G} fails:
    - matching: {H,J} . . .
  - if {A,B} succeeds and {F,G} fails:
    - . . .

Conclusions/Further work

- Very difficult problem
  - can we solve realistic cases?
  - how will practitioners react to the solution?
    - each solution may have an exponential number of steps
  - $\rightarrow$ example
- How to deal with multiple *agents*
  - *e.g.*, each agent may be an EU country
- To do: *extend* to cycles of size $k > 2$