CPS 196.2

Kidney exchanges

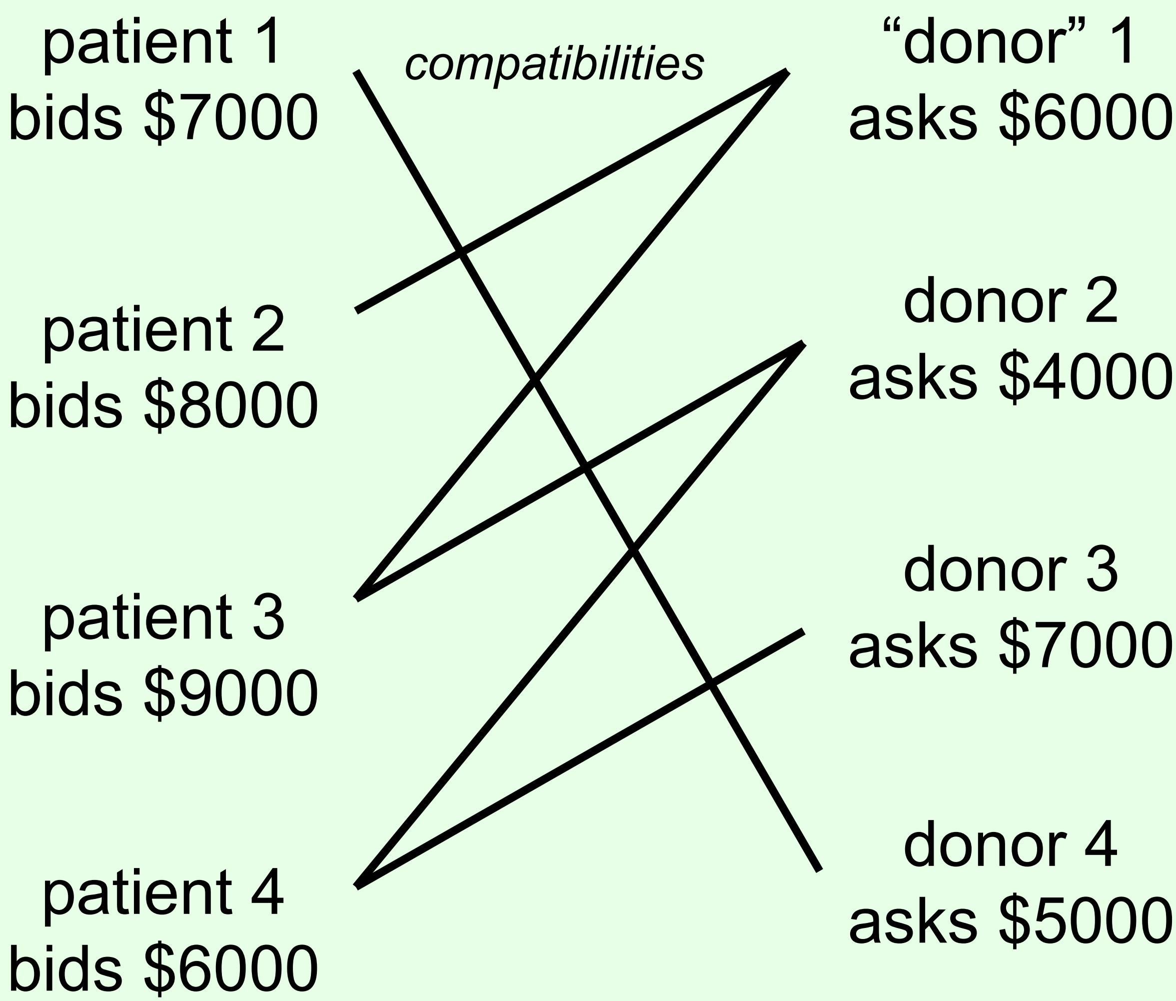
(largely follows Abraham, Blum, Sandholm 2007 paper)

Vincent Conitzer
conitzer@cs.duke.edu

Kidney transplants

- Kidneys filter waste from blood
- Kidney failure results in death in months
- Dialysis: regularly get blood filtered in hospital using external machine
  - Low quality of life
- Preferred option: kidney transplant
  - Cadaver kidneys
  - Donation from live patient (better)
- Must be compatible
- Shortage of kidneys…

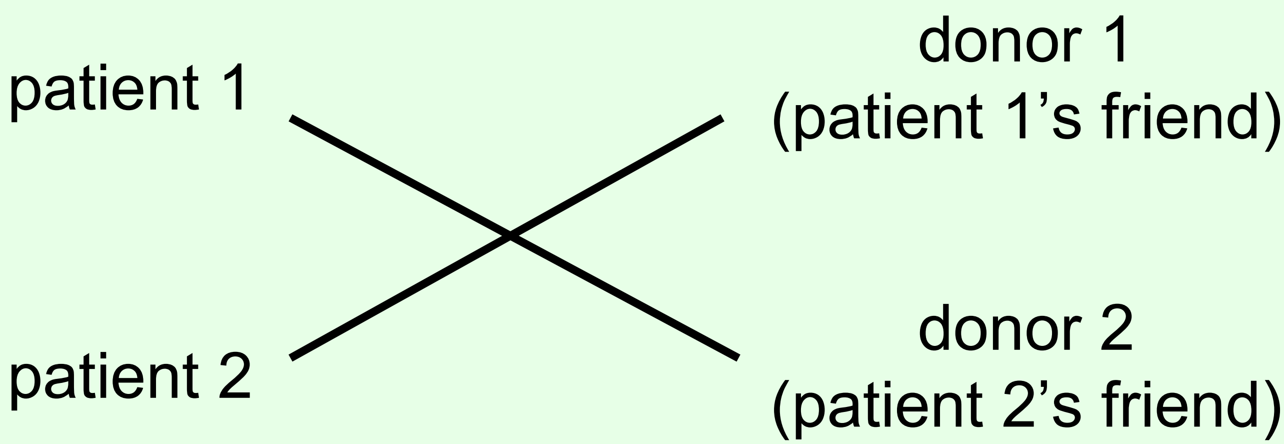
An imaginary kidney exchange with money



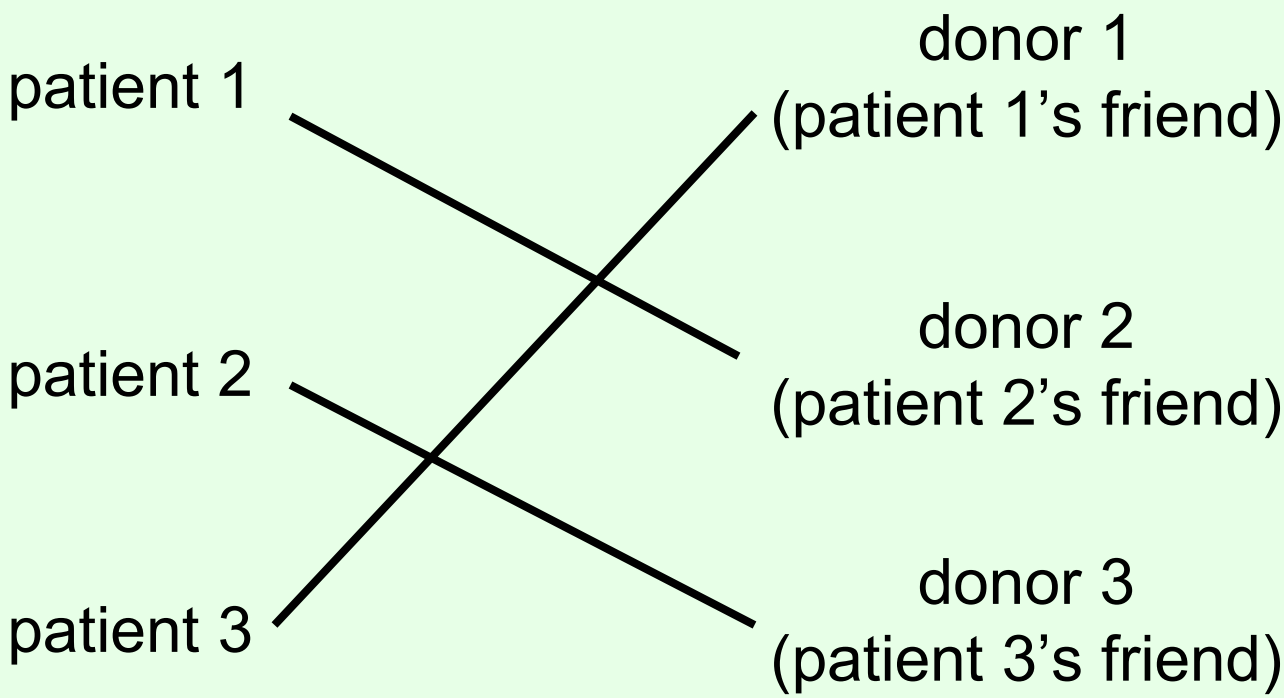
Selling kidneys is illegal!

- Large international black market
  - Desperate people on both ends…
- What can we do legally?

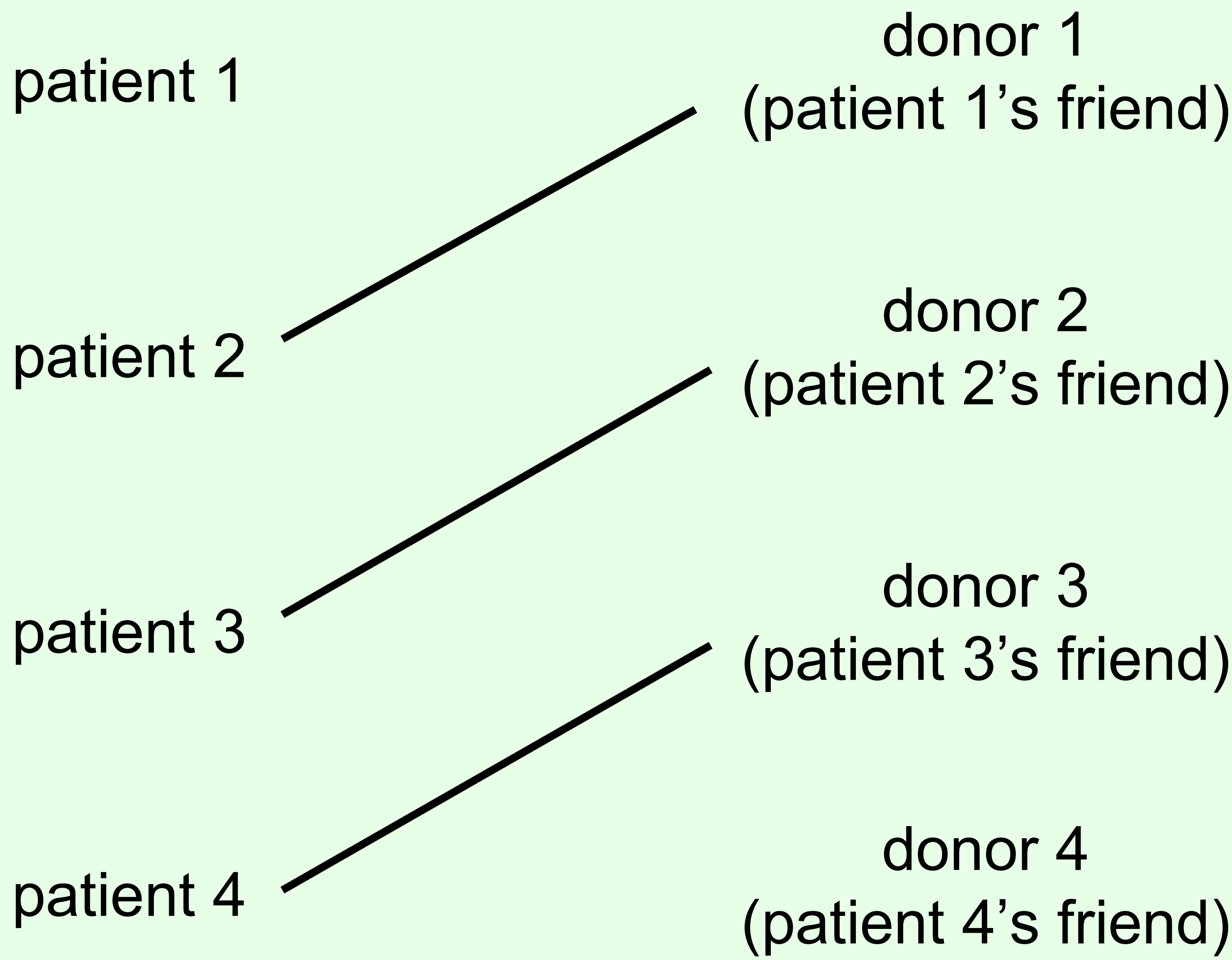
Kidney exchange



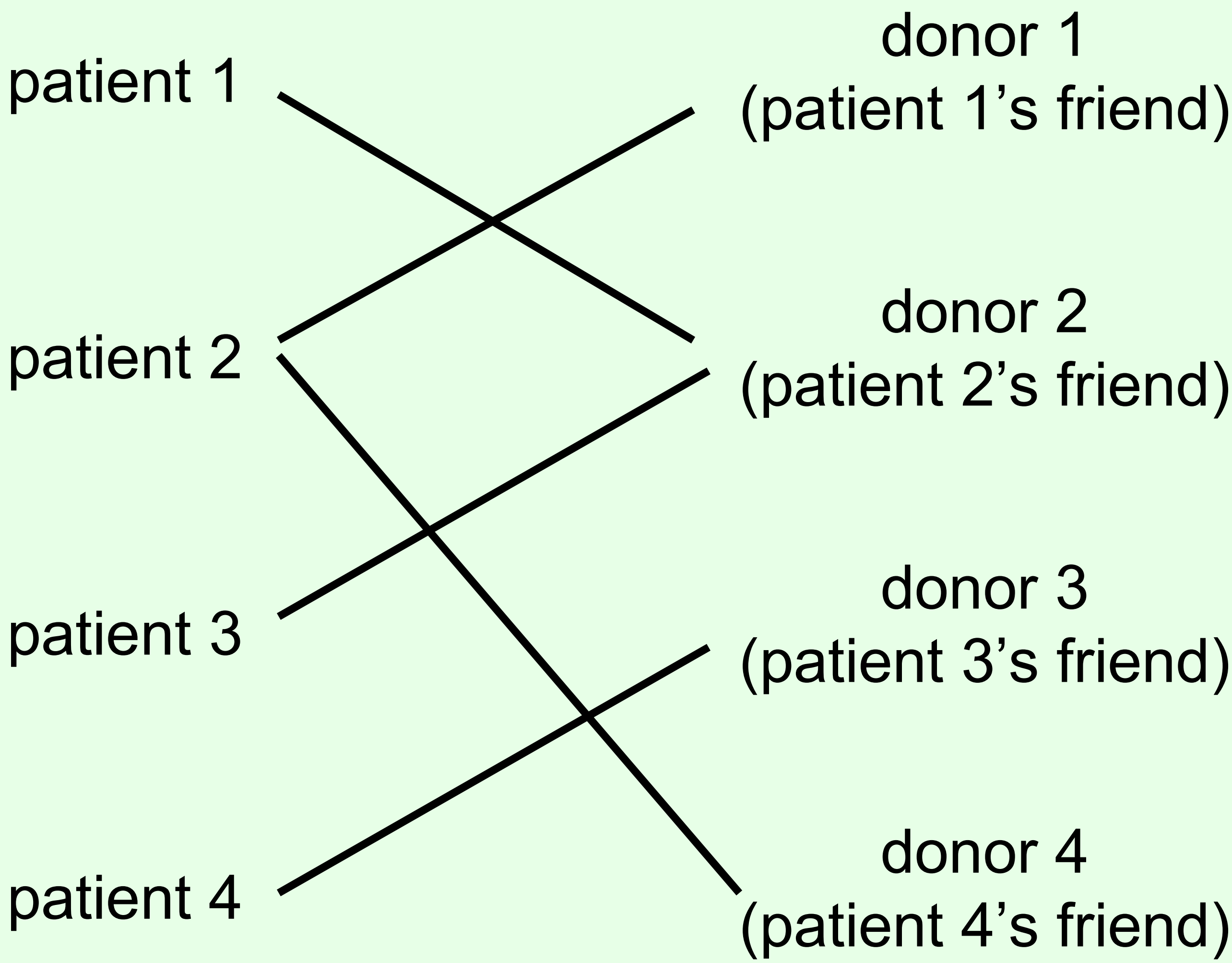
Kidney exchange (3-cycle)



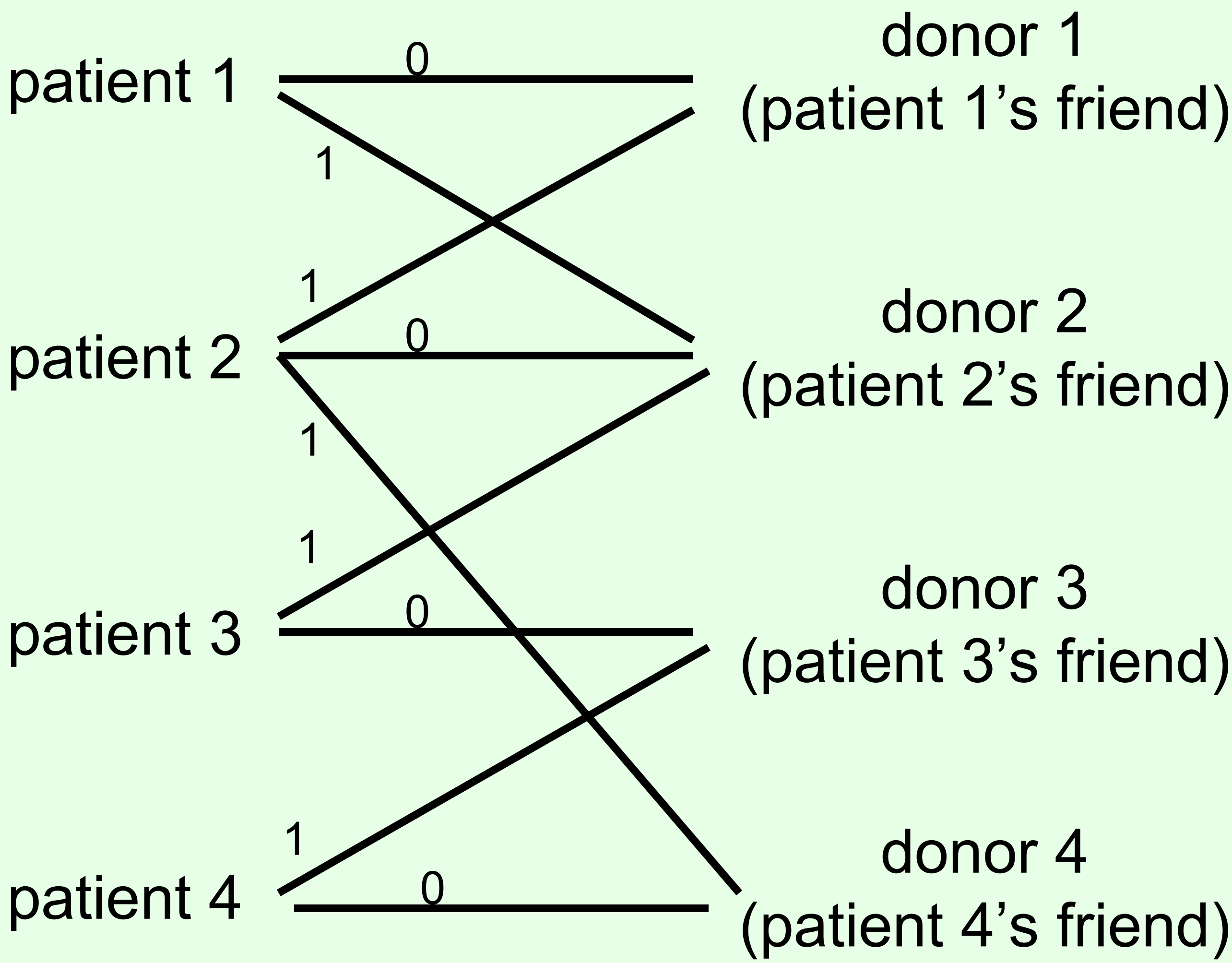
Another example



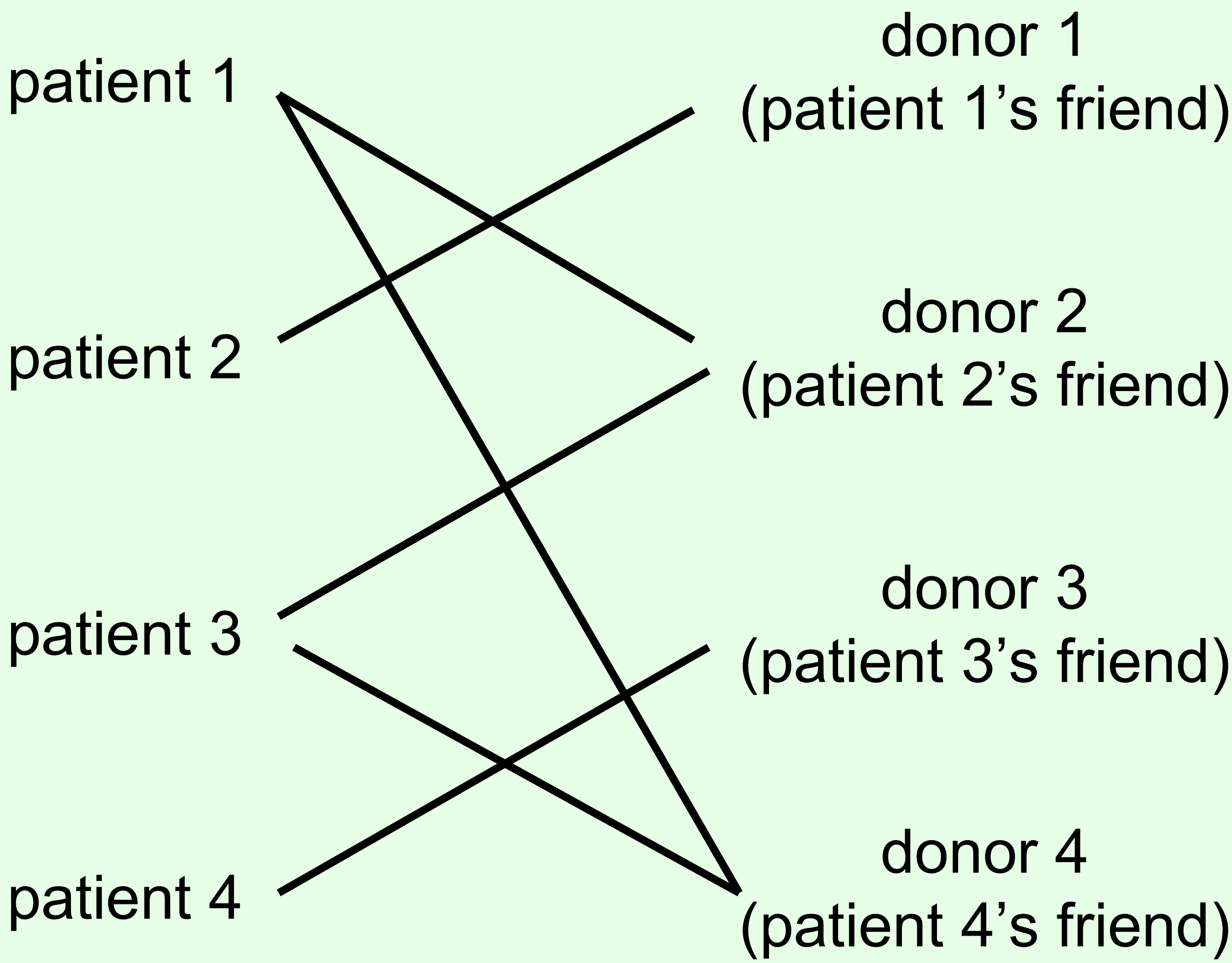
More complex example



Solving kidney exchange as maximum weighted bipartite matching



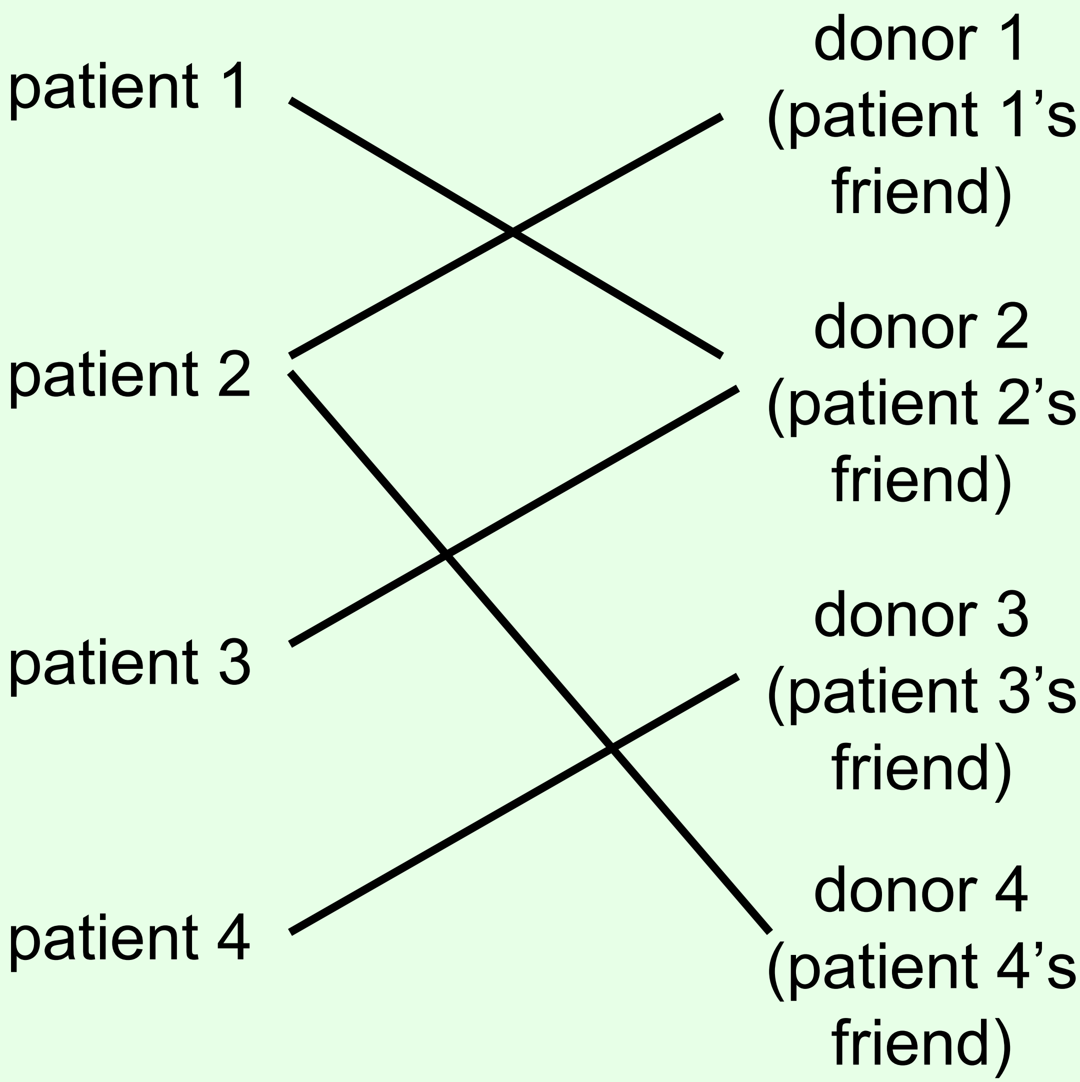
Which solution is better?

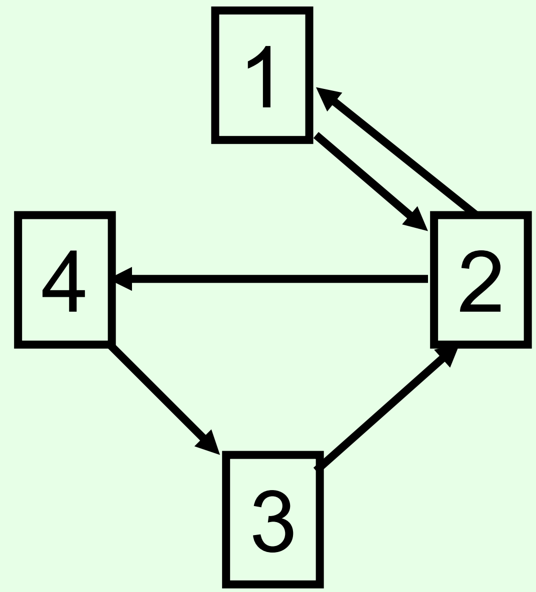


Long cycles are impractical

- All patients in a cycle must be operated on simultaneously
  - Otherwise donor can wait for friend to receive kidney, then back out
  - Contracts to donate an organ not binding
- If last-minute test reveals incompatibility, whole thing falls apart
- Require each cycle has length at most k

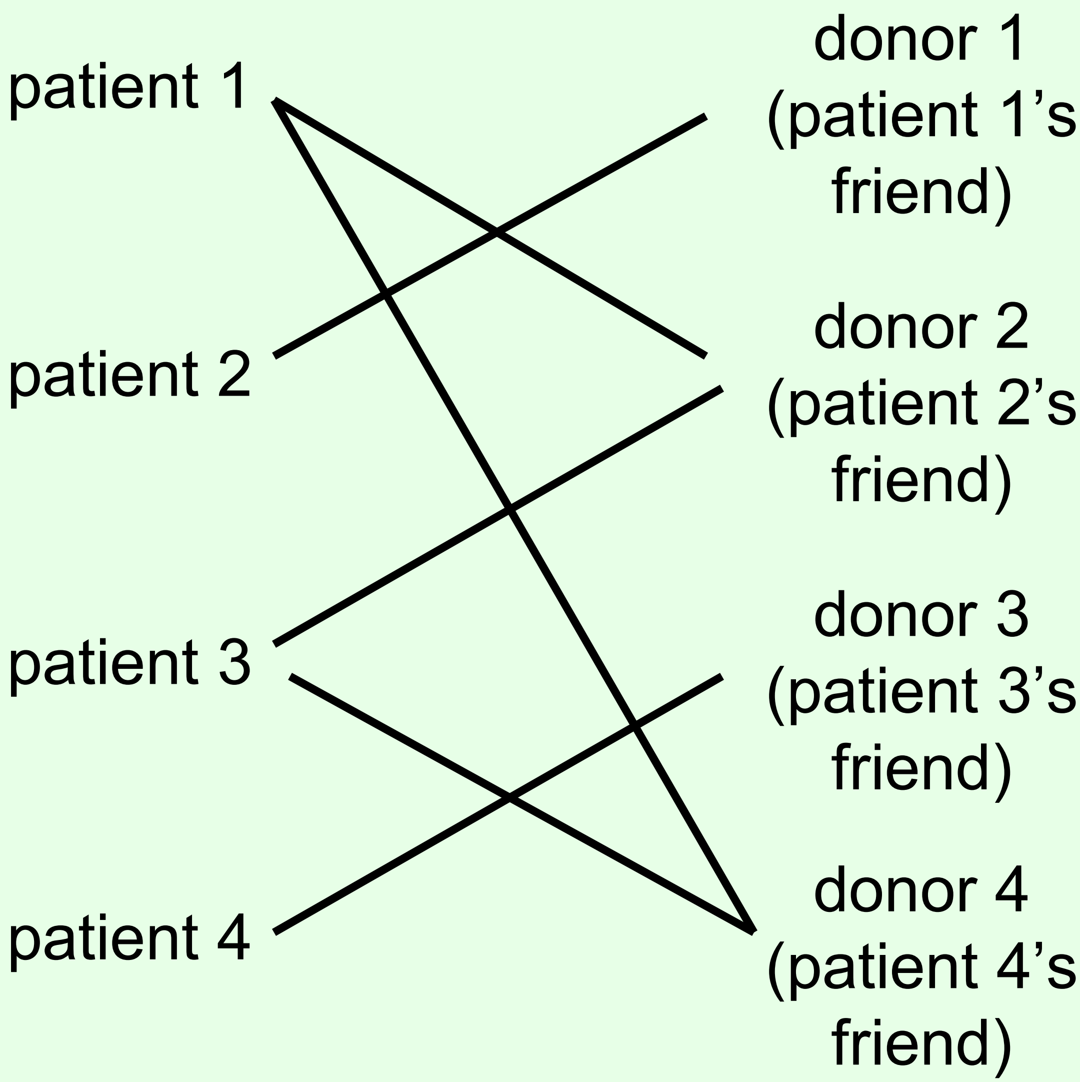
Different representation

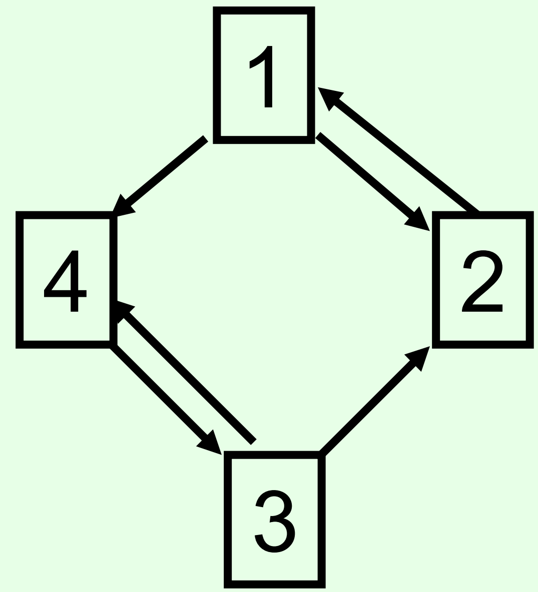




edge from i to j =
patient i wants
donor j's kidney

Different representation

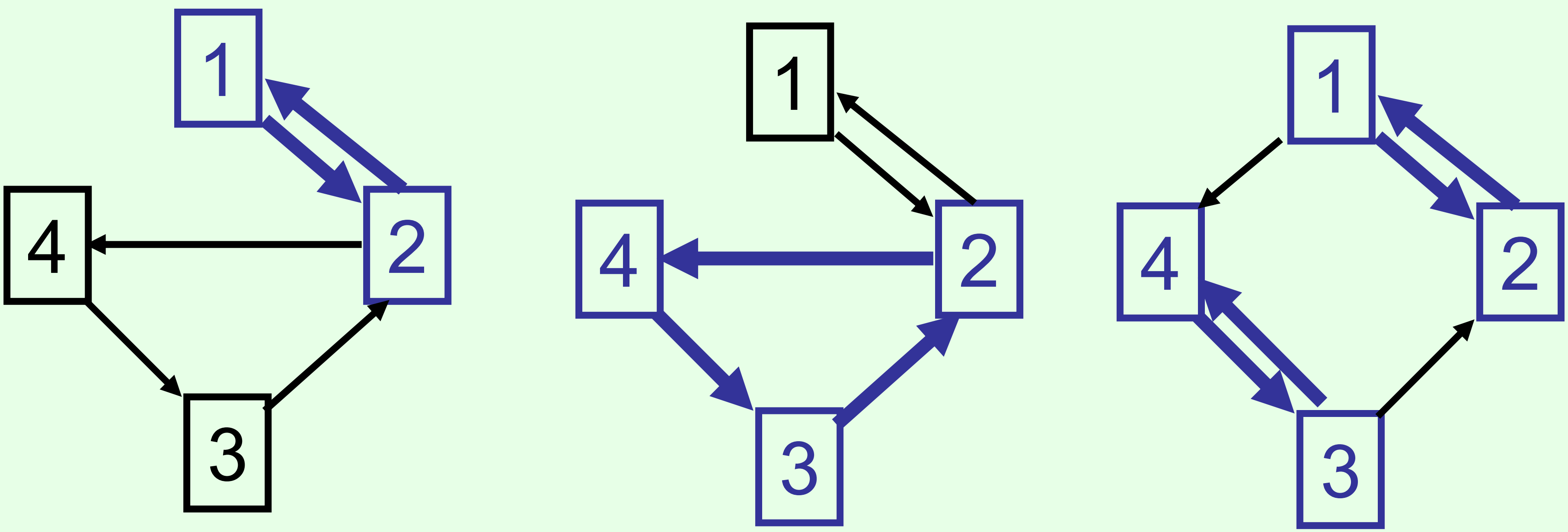




edge from i to j = patient i wants donor j's kidney

Market clearing problem

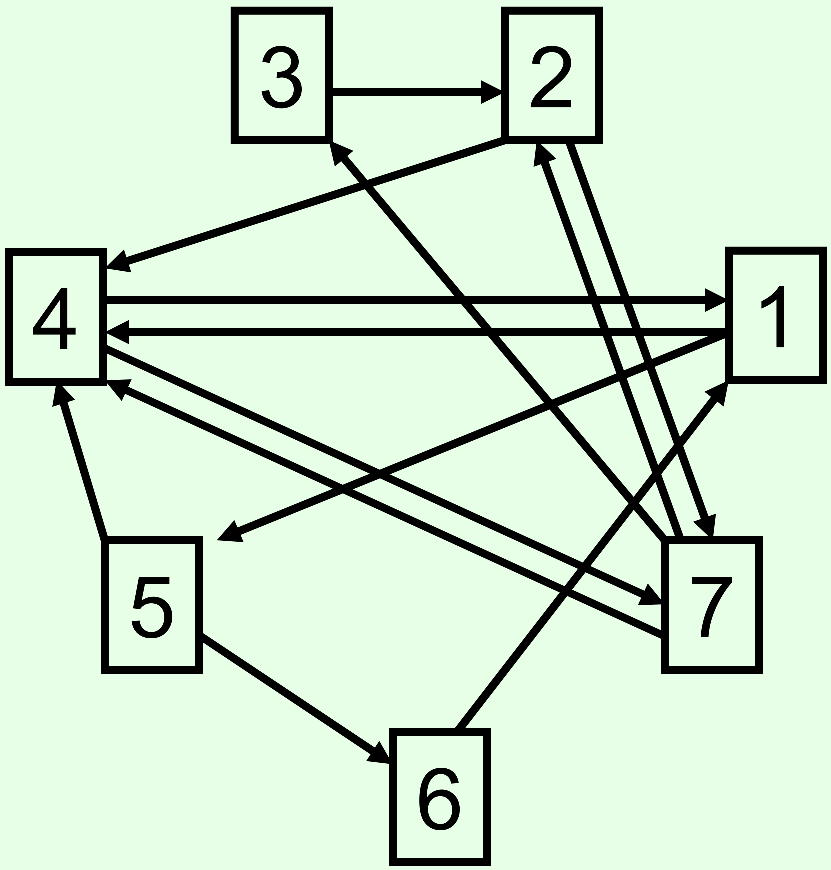
- Try to cover as many vertices as possible with (vertex-)disjoint cycles of length at most k



k=2 k=3 k=2,3

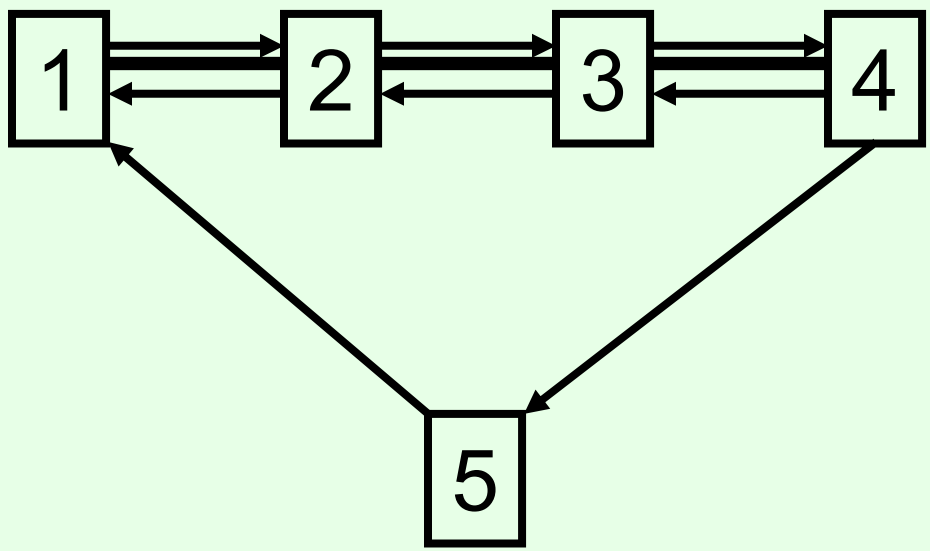
Market clearing problem

- Try to cover as many vertices as possible with (vertex-)disjoint cycles of length at most k



Special case: k=2

- If edges go in both directions, replace by undirected edge
- Remove other edges



- Maximum matching problem!

Complexity

- k = 2: in P by maximum matching
- k = number of vertices (no constraint): in P by maximum weighted bipartite matching
- k = 3, 4, 5, …: NP-hard!

An integer programming formulation

- For each edge from i to j, make a binary variable x_{ij}
  - 1 if i gets j's kidney, 0 otherwise
- maximize $\Sigma_{ij}x_{ij}$
- subject to:
- for every i: $\Sigma_j x_{ij} = \Sigma_j x_{ji}$
  - (number of kidneys received by i = number of kidneys given by i)
- for every j: $\Sigma_i x_{ij} \leq 1$
  - (j gives at most 1 kidney)
- for every path $i_1\ i_2 \ldots i_k\ i_{k+1}$ with $i_1 \neq i_{k+1}$: $\Sigma_{1 \leq j \leq k} x_{i_j i_{j+1}} \leq k-1$
  - (no path of length k that doesn't end up where it started, hence no cycles greater than k)

Another integer programming formulation
(turns out better)

- For each cycle c of length at most k, make a binary variable x_c
  - 1 if all edges on this cycle are used, 0 otherwise
- maximize $\Sigma_c|c|x_c$
- subject to:
- for every vertex i: $\Sigma_{c:\ i\ \text{in}\ c} x_c \leq 1$
  - (every vertex in at most one used cycle)

Program size

- Even for small k, number of paths/cycles is too large in reasonably large exchanges
- Solution: generate constraints/variables on the fly during solving
  - Constraint/column generation

Another integer program (not in paper)

- Say an "event" is a set of simultaneous operations
- Denote events by $t = 1, \ldots, T$ (how big should T be?)
- For each edge from i to j, for each t, make a binary variable x_{ijt}
  - 1 if i gets j's kidney in event t, 0 otherwise
- maximize $\Sigma_{i,j,t} x_{ijt}$
- subject to:
- for every i, t: $\Sigma_j x_{ijt} = \Sigma_j x_{jit}$
  - (number of kidneys received by i in event t = number of kidneys given by i in event t)
- for every j: $\Sigma_{i,t} x_{ijt} \leq 1$
  - (j gives at most 1 kidney overall)
- for every t: $\Sigma_{i,j} x_{ijt} \leq k$
  - (at most k operations per event)

Other applications

- Barter exchanges: agents want to swap items without paying money
- Peerflix (DVDs)
- Read It Swap It (books)
- Intervac (holiday houses)
- National odd shoe exchange
  - People with different foot sizes
  - Amputees