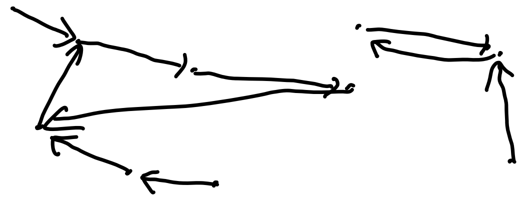
4/19/22: Mechanisms Without Money

House Allocation:

- Players $[n]$, each has a house
- Each player has private total ordering (preferences) over all houses
- Can we (re)allocate houses to make players happier?
- "Bid" is an ordering

Top Trading Cycle:

- Let $A = [n]$
- while $A \neq \emptyset$:
  - For each $i \in A$, let $f(i)$ be favorite house belonging to agent in A
  - Directed graph $G = (A, \{(i, f(i))\}_{i \in A})$



e.g.

Every vertex has outdegree 1

$\Rightarrow$ ≥ 1 directed cycle,

every vertex in ≤ 1 directed cycle

- For each directed cycle, reallocate: every vertex gives house to predecessor
- Remove all agents who were reallocated from A

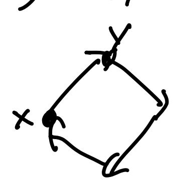
Easy properties: will terminate with feasible allocation, in polynomial time

Incentive Compatibility:

Truthful bidding $\Rightarrow$ no worse off: (get house at least as good as starting)

Allocated along cycle, could have loops

e.g.



Truthful bidding is dominant:

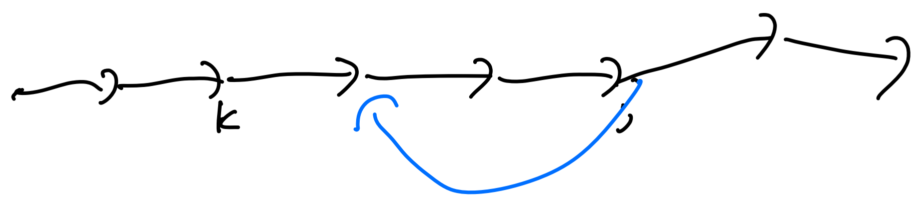
Fix agent $i \in [n]$, all other bids.

If i bids truth: $N_i := \{$agents allocated in iteration $i\}$

Let i^* s.t. $i \in N_{i^*}$

Claim 1: i gets favorite house not in $N_1 \cup \ldots \cup N_{i^*-1}$

Claim 2: If k has path to i in some iteration, then that path in all graphs until i allocated (iteration i^*)



$\Rightarrow$ if i lies, creates a cycle with nodes P that have path to i

$\Rightarrow P \cap (N_1 \cup N_2 \cup \ldots \cup N_{i^*-1}) = \emptyset$

$\Rightarrow$ lying doesn't help!

Quality: No utilities or money: how can we even define quality of an allocation?

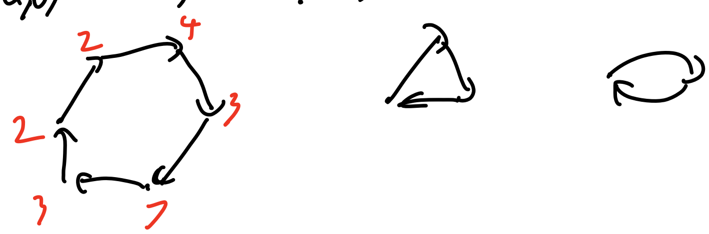
Def: Blocking coalition of an allocation: set of agents who can pull out from allocation and reallocate among themselves so no one worse off, ≥ 1 better off

Def: A core allocation is an allocation with no blocking coalitions

Thm: TTC returns a core allocation

Pf: Let $S \subseteq [n]$ be some coalition, and suppose they reallocate among themselves.

$\Rightarrow$ graph of disjoint directed cycles:
(u,v) means v gives house to u



If cycle all in same N_i:

All in cycle would get favorite out in $N_1 \cup \cdots \cup N_{i-1}$ from TTC

If cycle has nodes from different N_is:

$u \in N_i \longrightarrow v \in N_j \quad j > i$

$\Rightarrow$ u worse off!

Thm: TTC returns only core allocation

Pf: Induction.

TTC: every agent in N_1 gets favorite house

$\Rightarrow$ Every core allocation agrees with TTC on N_1, or else N_1 would be blocking coalition

Suppose some core allocation agrees with TTC on $N_1 \cup \dots \cup N_{i-1}$

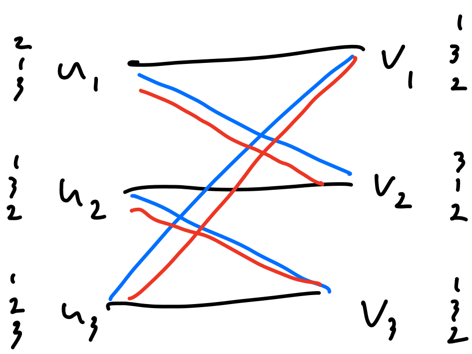
$\Rightarrow$ must agree with TTC on N_i, or else N_i a blocking coalition

Stable Matching:

- Two sets of agents U, V $\quad |U| = |V| = n$
- Each agent has total ordering over other set

<u>Def</u>: A stable matching is a perfect bipartite matching (bijection $\pi : U \to V$) s.t. if u, v not matched then either:

- u prefers $\pi(u)$ to v, or
- v prefers $\pi^{-1}(v)$ to u



<u>Q</u>: - Do stable matchings always exist?

- If yes, can we compute them?
- Incentive-Compatible?

Proposal/ Deferred Acceptance Algorithm:

while there is an unmatched $u \in U$:

- u sends a proposal to favorite $v \in V$ that has not yet rejected u
- If v unmatched, match u and v
- If v matched to $u' \in U$, v chooses to match to favorite of u, u', rejects the other

Lemma: At most n^2 iterations

Pf:

Every iteration is new proposal

At most n^2 different proposals

$\Rightarrow \leq n^2$ iterations

Lemma: Terminates with a perfect matching

Pf: Always matching, why perfect?

Sps $u \in U$ not matched

$\Rightarrow$ rejected by all $v \in V$

$\Rightarrow$ every $v \in V$ gets at least one proposal

$\Rightarrow$ every $v \in V$ matched at end

$\Rightarrow$ every $x \in U$ matched $\Rightarrow\Leftarrow$

<u>Lemma</u>: The perfect matching is stable

<u>Pf</u>: Suppose $u \in U$, $v \in V$ not matched.

If u never proposed to v:
  u matched to someone it prefers over v

If u did propose to v:
  v matched to someone it prefers over u

Is this a "good" stable matching?

<u>Def</u>: For $u \in U$, let $h(u)$ = favorite node in V that u is matched to in <u>any</u> stable matching

Thm: In matching from algorithm, **every** $u \in U$ is matched to $h(u)$

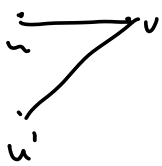
Pf: Let $R = \{(u,v) : v \text{ rejects } u \text{ in algorithm}\}$

WTS: If $(u,v) \in R \Rightarrow$ no stable matching matches u and v

Induction (algorithm invariant):

At beginning, $R = \emptyset$

Consider iteration where u rejected by v in favor of u' (add (u,v) to R)



$\Rightarrow v$ prefers u' to u

By def of algorithm $(u',v') \in R$ $\forall v'$ that u' prefers to v

$\Rightarrow$ by induction, no stable matching where u' does better than v.

So sps $\exists$ stable matching with (u,v)

$\Rightarrow u'$ is doing worse than v, and v is doing worse than u'

$\Rightarrow$ not stable!