Algorithms

DATE : 24/02/2022

L = LEMMA
P = PROOF

GALE - SHAPELEY ALGORITHM

- Initially, each $a_i \in A$, and each $b_j \in B$, is FREE
- While there exists some FREE a_i that has not yet proposed to each $b_j \in B$
- Let a_i be a FREE person that has not proposed to each $b_j \in B$
- Let $B' \subseteq B$ be the set of b_j such that a_i has not yet proposed to
- Let $b_j \in B'$ be the person from B' that a_i likes the most

$$\left.\begin{aligned} a_1 &: b_1 > b_2 > b_3 \\ B' &= \{ b_2, b_3 \} \end{aligned}\right\} \text{ } b_2 \text{ is the most preferred in the } B' \text{ set}$$

- IF b_j is FREE :
  - MATCH UP a_i and b_j // a_i & b_j get engaged
  - a_i and b_j are not free anymore
- ELSE :
  - Suppose that b_j is engaged to a_K
  - IF b_j likes a_K more than a_i
    - a_i REMAINS FREE
  - ELSE :
    - the match between b_j and a_K is broken
    - a_i and b_j are matched up
    - a_K becomes FREE
- RETURN THE FINAL LIST OF "MATCHES" AS THE MATCHING.

NOTE: This still doesn't define whether the final matching is perfect. Proof needed.

TECHNICAL OBSERVATIONS

L1: Each $b \in B$ remains matched / engaged from the first time she gets a proposal until the end of the execution

↑

PROOF: When "b" gets the first proposal, she becomes engaged. (Since it's her first time, NO REFUSE)

From then onwards she might get other proposals. She might either:

- accept
- reject

SHE WILL ALWAYS BE ENGAGED

If she accepts one, she'll switch partners (but she'll remain engaged);

If she rejects, she'll keep her previous partner.

L2: The engagements of the generic $b \in B$ get better (from her perspective) over the time

↑

P: b changes partner only if she gets a proposal from a better then her current one.

MONOTONE PROPERTY (keeps getting better)

THE A SIDE HAS A DIFFERENT FATE

L3: For each sequence of proposals made by a decreases in quality over time.

↑

P: TRIVIAL (by the algorithm's def)

NOTE: Theorem → more important than "lemma"

Theorem(T): The algorithm terminates after at most n^2 iterations.

↑

Proof (P): Each a_i can propose to at most $|B| = n$ people from B. In each iteration of the algorithm, some a_i proposes to some $b_j \in B$ that he had not yet proposed to earlier. ↓

Therefore, there can be at most $|A| \cdot |B| = n^2$ proposals, and iterations

---

In general, people look for some "quantities" to bound the runtime of an algorithm

L4: IF $a \in A$ is FREE at some point in the execution, then there must exist some $b \in B$ to which a has not yet made a proposal.

TECHNICAL OBS.

(THIS LEMMA LEADS TO THE FACT THAT WE WILL HAVE A PERFECT MATCHING IN THE END)

P: By contradiction, suppose that, at some point, $a^* \in A$ is FREE and he has proposed to everyone from B

By L1, each $b \in B$ remains engaged from the first proposal she gets, until the end. Thus, for a^* to remain FREE after $n = |B|$ proposals, it must be that, at the time of his last proposals, each $b \in B$ was engaged.

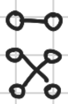
But, recall that $|A| = |B| = n$.

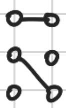
For each $b \in B$ to be engaged, it must be that each $a \in A$ must be engaged. Thus, it is IMPOSSIMPLE that a^* is FREE.

> **THIS IS A PROOF BY CONTRADICTION**
>
> - Basically, you start from an assumption and end up contradicting

L5: The algorithm returns a perfect matching

P: When we match a to b, if b was already matched, then we break up the current engagement of b. Thus, the current matches from matching


PERFECT MATCHING

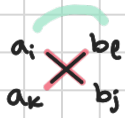

MATCHING

Suppose, by contradiction, that in the end $a \in A$ is FREE. Then, a has proposed to each $b \in B$. But, this contradicts L4. Thus, in the end, no $a \in A$ is FREE and, also, no $b \in B$ is FREE. ($|A| = |B|$ and the returned structure is a matching). Thus, the algorithm returns a perfect matching.

THEOREM: The algorithm returns a stable matching

PROOF: By L5, the algorithm returns a perfect matching (each person is matched to exactly one other person).

By contradiction, suppose that there exist two pairs in M, $\{a_i, b_j\}$, $\{a_k ; b_\ell\}$ that are UNSTABLE



$a_i \cdots\!> b_\ell \cdots\!> b_j$

$b_\ell \cdots\!> a_i \cdots\!> a_k$

Then, a_i prefers b_ℓ to b_j, and b_ℓ prefers a_i to a_k. By the algorithm, a_i's last proposals was to b_j

Now, let us consider two cases:

- a_i did not propose to b_ℓ before b_j. Then, a_i did not ever propose to b_ℓ (b_j is a_i's last proposal). But then, a_i prefers b_j to b_ℓ CONTRADICTION

- a_i proposed to b_ℓ before b_j. Then, since b_ℓ ended up with a_k, and since L2 entails that b_ℓ's partners improve over time, it must be that b_ℓ prefers a_k to a_i. CONTRADICTION

Therefore, no unstable pairs exist - Thus, M is a stable matching.

DEF: Let us say that $b_j \in B$ is a valid match for $a_i \in A$ if $\exists$ stable matching M such that $\{a_i, b_j\} \in M$

best partner from a_i's perspective

DEF: Let 'best(a_i)', for $a_i \in A$, be the valid match $b^* \in B$ of a_i, that a_i likes the best

THEOREM: The G-S algorithm returns $M = \{\{a_i, \text{best}(a_i)\} \mid a_i \in A\}$

THEOREM: " " " " $M = \{\{b_j, \text{worst}(b_j)\} \mid b_j \in B\}$

(This is a bit unfair since one gets the best and the others get the worst, all in the stable matching)