ANALYSIS OF ALGORITHMS

LECTURE 12 : STABLE MARRIAGE (PART 2)

BASED ON SECTION 4.3

READ THE BOOK

- The book as a lot of information which I am not covering in the lecture, and you can be tested on anything that's in the book!

HOW DO WE FIND STABLE MARRIAGE SETS?

- Last week we talked about how we can test that a set of marriages is stable
- This week we talk about how to get a set of stable marriages
- The way we will talk about is called the Gale-Shapley algorithm
- Works as follows: (you may find this familiar)
  - Boys propose to girls in order of preference
  - If a girl is not currently engaged, she will accept the proposal
  - If she is engaged, she will decide whether she likes the boy who is proposing more than she likes her fiancé – If she does, she will leave her fiancé and accept the proposal and if not, she will reject the proposal.

WHAT DO WE NEED TO KEEP TRACK OF?

- Gale-Shapley algorithm
  - Boys propose to girls in order of preference
  - If a girl is not currently engaged, she will accept the proposal
  - If she is engaged, she will decide whether she likes the boy who is proposing more than she likes her fiancé – If she does, she will leave her fiancé and accept the proposal and if not, she will reject the proposal.
- What data do we need to keep track of?
  - Preference arrays
  - Marriage array – can keep track of fiancés
  - Proposal array – keeps track of who each boy has already proposed to

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	
1	
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	
1	
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	2
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	2
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	2
2	
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	2
2	3
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	0
1	2
2	3
3	

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	
1	2
2	3
3	0

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	
1	2
2	3
3	0

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	
1	2
2	3
3	0

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

LET'S RUN THROUGH THE ALGORITHM

Matches

M	W
0	1
1	2
2	3
3	0

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

Rank

man	0	1	2	3
0	0	2	1	3
1	2	0	1	3
2	2	3	0	1
3	0	1	3	2

Rank

woman	0	1	2	3
0	3	1	2	0
1	3	0	2	1
2	1	3	2	0
3	1	0	2	3

THAT SEEMS NICE… BUT DOES IT ALWAYS WORK?

- Firstly, does the algorithm terminate?
  - Well how would it not?
  - Since engagements can be created and destroyed, maybe it will just never end

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

THAT SEEMS NICE… BUT DOES IT ALWAYS WORK?

- Firstly, does the algorithm terminate?
  - Well how would it not?
  - Since engagements can be created and destroyed, maybe it will just never end
  - But note that boys just run down their preference lists, so every boy can only propose n times
  - So there will be at most n^2 proposals
  - So the algorithm will terminate

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

THAT SEEMS NICE… BUT DOES IT ALWAYS WORK?

- Secondly, when it terminates, is everyone married?
  - Note that for a boy to end up unattached, he would have to be rejected by every girl
  - The only way a girl would reject him is if she is already with someone better
  - So every girl was attached at the point he proposed
  - But note that if a girl is attached, it's impossible for her to become unattached – she only ends an engagement to enter a better one
  - So that means every girl is engaged at the end
  - But there's an equal number of boys and girls, so there can't be an unattached boy

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

THAT SEEMS NICE… BUT DOES IT ALWAYS WORK?

- Thirdly, are the produced marriages always stable?
  - Let (b_x, g_y) be a marriage in the set
  - Let g_z be a girl that b_x likes more
  - b_x didn't end up with g_z, so she must have rejected him at some point, and is then paired with some boy b_a
  - Now the thing is she must like b_a more than b_x or she would have accepted his proposal instead
  - So that means that b_x can't run away with g_z, and this is true analogously for all g_z
  - b_x is an arbitrary choice, so it's true for all boys

- While there is some unmatched boy b
  - Let g be the first girl that b has not proposed to
  - If g is unattached
    - Match b and g
  - else if g prefers b to her fiancé b'
    - Unmatch b' and g
    - Match b and g
  - else leave b unpaired

OPTIMALITY

- Here's the slightly disturbing thing about this algorithm
- It results in the best possible set of stable marriages for boys! (If girls propose, it results in the best possible set of marriages for girls)
  - Note that this doesn't mean boys end up with their favourite girl because those marriages would not necessarily be stable
  - But from all possible STABLE sets of marriages, the algorithm results in the BEST possible set of marriages for boys
  - Note that this doesn't just mean that it's the best on average, it's actually the best for every boy independently as well
- And we can prove it!

OPTIMALITY

- RTP: Male oriented Gale Shapley results in the best possible set of stable marriages for boys – the male optimal set
- (Please read the proof of theorem 4.5 in the book for a slightly more formal treatment, I'm going to be a bit informal here which is bad for a proof)
- We're going to use proof by contradiction.

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications…
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If A is true
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$
- So let's say I know nothing about the truth of statements A through F
- If I know that G is false, then A must be false, because if it were true, G would be true

SIDE NOTE : PROOF BY CONTRADICTION

- Let's say we have a chain of implications – If G is false
- $A \Rightarrow B \Rightarrow C \Rightarrow D \Rightarrow E \Rightarrow F \Rightarrow G$
- So let's say I know nothing about the truth of statements A through F
- If I know that G is false, then A must be false, because if it were true, G would be true
- So how we do a proof by contradiction is we start with some A that we're trying to prove is false (our assumption)
- And then we construct a chain to some G that we know is false

PROOF OF MALE OPTIMALITY

- In this case our assumption is that GS is not Male Optimal
- Let the set of marriages produced by GS be M_{GS}
- Let the M_x be a set of stable marriages in which some males are better off than in M_{GS}
- Note that M_x is not produced by GS, we have no restriction on how this set of marriages came about
- Now, when we were producing M_{GS}, some boys must have been rejected by their partners in M_x (Because they propose in order of preference)
- Let b_i be the first such boy, and let his partner in M_x be g_j
- Now, g_j would only reject him because she is with someone better, lets call him b_k

PROOF OF MALE OPTIMALITY

- So g_j likes b_k more than b_i
- b_k has proposed to g_j already, because that's why she rejected b_i
- Now, we said that b_i was the first boy rejected by his partner in M_x
- So that means that b_k has not at this point been rejected by his partner in M_x
- Which means he hasn't proposed to her yet
- Which means that b_k likes g_j more than his partner in M_x (he proposes in order of preference)
- But the thing is we already know that g_j likes b_k more than she likes b_i (her partner is M_x)
- That means that M_x is unstable as b_k would run off with g_j
- But earlier we said : Let M_x be a set of **stable** marriages in which some males are better off than in M_{GS}
- CONTRADICTION
- So there is no other stable marriage set in which any boy is with a better partner

OPTIMALITY

- Here's a slightly more depressing result
- The male optimal version of the Gale Shapley algorithm produces the worst possible set of marriages for girls
- Even more depressing, this is just a consequence of it producing the best possible set of marriages for boys
- The best set of marriages for boys is automatically the worst possible set of marriages for girls
- I have asked for a proof of this in every exam for the last five years or so, and I've never taught it in class, so look it up, it is very likely to be worth some marks ☺