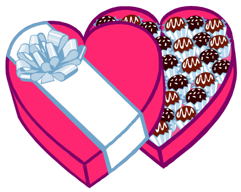
Love, Marriage, and Lying

Lecture P4



Overview

Use arrays to solve interesting problem.

Understand mathematics of marriage proposals.

- Who benefits more, the men or women?
- Who should misrepresent their feelings?

Enjoy yourself today!

Standard disclaimer.

Stable Marriage Problem

Problem: Given N men and N women, find a “suitable” matching between men and women.

- Participant have ordered preference list of members of opposite sex.
- Each man lists women in order of preference from best to worst.

Men’s Preference List

Man	0^{th}	1^{st}	2^{nd}	3^{rd}	4^{th}
Victor	Bertha	Amy	Diane	Erika	Clare
Wayne	Diane	Bertha	Amy	Clare	Erika
Xavier	Bertha	Erika	Clare	Diane	Amy
Yancey	Amy	Diane	Clare	Bertha	Erika
Zeus	Bertha	Diane	Amy	Erika	Clare

best — worst

Stable Marriage Problem

Problem: Given N men and N women, find a “suitable” matching between men and women.

- Participant have ordered preference list of members of opposite sex.
- Each man lists women in order of preference from best to worst.
- Each woman lists men in order of preference.

Women’s Preference List

Woman	0^{th}	1^{st}	2^{nd}	3^{rd}	4^{th}
Amy	Zeus	Victor	Wayne	Yancey	Xavier
Bertha	Xavier	Wayne	Yancey	Victor	Zeus
Clare	Wayne	Xavier	Yancey	Zeus	Victor
Diane	Victor	Zeus	Yancey	Xavier	Wayne
Erika	Yancey	Wayne	Zeus	Xavier	Victor

best — worst

Stable Marriage Problem

Problem: Given N men and N women, find a "suitable" matching between men and women.

- Everyone is matched monogamously (perfect matching).
  - each man gets exactly one woman
  - each woman gets exactly one man
- Stable: no incentive for some pair of participants (or coalition) to undermine assignment by joint action.
  - an unmatched pair (m,w) is UNSTABLE if man m would prefer woman w to his wife, and w would prefer m to her husband
  - unstable pair could each improve by dumping spouses and eloping

STABLE MARRIAGE = perfect matching with no unstable pairs.
(Gale and Shapley, 1962)

Example

Men's Preference List

Woman	0th	1st	2nd
Xavier	A	B	C
Yancey	B	A	C
Zeus	A	B	C

Women's Preference List

Woman	0th	1st	2nd
Amy	Y	X	Z
Bertha	X	Y	Z
Clare	X	Y	Z

Lavender assignment is a perfect matching.
Is it stable?

Example

Men's Preference List

Woman	0th	1st	2nd
Xavier	A	B	C
Yancey	B	A	C
Zeus	A	B	C

Women's Preference List

Woman	0th	1st	2nd
Amy	Y	X	Z
Bertha	X	Y	Z
Clare	X	Y	Z

Green assignment is a stable matching.

Example

Men's Preference List

Woman	0th	1st	2nd
Xavier	A	B	C
Yancey	B	A	C
Zeus	A	B	C

Women's Preference List

Woman	0th	1st	2nd
Amy	Y	X	Z
Bertha	X	Y	Z
Clare	X	Y	Z

Orange assignment is also a stable matching.

Stable Roommate Problem

Not obvious that stable marriage exists.

Consider related "stable roommate problem."

- 2N people.
- Each person ranks others from 0 to 2N-2.
- Assign roommate pairs so that no unstable pairs.

Preference List

	0th	1st	2nd
Adam	B	C	D
Bob	C	A	D
Chris	A	B	D
Doofus	A	B	C

No perfect matching is stable.

For all 3 possible perfect marriage, can always find unstable pair.

E.g., A-C forms unstable pair in lavender marriage.

Existence

Surprising Fact:

- Unlike for stable roommate problem, one (or more) stable marriages exist for any input to problem.

How do we find one?

- Are there others?
- Which one is best for Zeus?
- Is there one that is best for all the men collectively? All the women?

Propose-And-Reject Algorithm

Formal (and intuitive) method that guarantees to find a stable marriage.

Repeat until no unmatched men

- An unmatched man m proposed to his favorite woman w to whom he has not already proposed.
- If w is unmatched, she accepts proposal from m (but can later dump him).
- Otherwise, if w prefers her current fiancé to m, she reject m outright.
- Otherwise, if she prefers m to her current fiancé, she dumps her fiancé and accepts the proposal from m.

Demo

Why Does Algorithm Work?

Observation 1. Men propose to their favorite women first.

Observation 2. Once a woman is matched, she never becomes unmatched. She only "trades up."

Fact 1. All men and women get matched.

- Suppose upon termination Zeus is not matched.
- Then some woman, say Amy, is not matched upon termination.
- By Observation 2, Amy was never proposed to.
- But, Zeus proposes to everyone, since he ends up unmatched. (contradiction)

Why Does Algorithm Work?

Observation 1. Men propose to their favorite women first.

Observation 2. Once a woman is matched, she never becomes unmatched. She only "trades up."

Fact 2. No unstable pairs.

- Suppose Zeus-Amy is an unstable pair, i.e., each prefers each other to spouse. (Zeus-Bertha, Yancy-Amy)
- Case 1. Zeus never proposed to Amy.
  - ⇒ Zeus must prefer Bertha to Amy (Observation 1)
  - ⇒ Zeus-Amy is stable. (contradiction)
- Case 2. Zeus proposed to Amy.
  - ⇒ Amy rejected Zeus (right away or later)
  - ⇒ Amy prefers Yancy to Zeus (women only trade up)
  - ⇒ Zeus-Amy is stable (contradiction)

Pseudocode

```
int marriages = 0;

while (marriages < N)
   find unmatched man m

   while (m unmatched)
      let w be man m's favorite women to
      whom he has not yet proposed

      if (w unmatched)
         m and w get engaged
         marriages++;
         break;

      if (w prefers m to current fiancé f)
         f now unmatched
         m and w get engaged
         break;

      else w rejects m
```


How to Represent Men and Women

Represent men and women as integers between 0 and N-1.

- 0 through N-1 since C array indices start at 0.
- Could use `struct` if we want to carry around more information, e.g., name, age, astrological sign.

How to Represent Marriages

Use array to keep track of marriages.

$$\textbf{wife}[m] = \begin{cases} w & \text{if man } m \text{ matched to woman } w \\ -1 & \text{if man } m \text{ unmatched} \end{cases}$$

$$\textbf{husb}[w] = \begin{cases} m & \text{if man } m \text{ matched to woman } w \\ -1 & \text{if woman w unmatched} \end{cases}$$

```
int wife[N];
int husb[N];

for (m = 0; m < N; m++)
  wife[m] = -1;
for (w = 0; w < N; w++)
  husb[w] = -1;
```

Filling in Some of the Code

find unmatched man

while (m unmatched)

if (w unmatched)
  m and w get engaged

f = current fiancé of w
f now unmatched
m and w get engaged

```
while (marriages < N)

   for (m = 0; wife[m] != -1; m++)
      ;

while (wife[m] == -1)
   let w be man m's favorite women to
   whom he has not yet proposed

   if (husb[w] == -1)
      husb[m] = w;
      wife[m] = w;
      marriages++;
      break;

   if (w prefers m to current fiancé f)
      f = husb[w];
      wife[f] = -1;
      husb[m] = w;
      wife[w] = m;
      break;
```


Representing the Preference Lists

Use 2D-array to represent preference lists.

- 2D-array is array of arrays.
- `mp[m][i] = w` if man m's i^{th} favorite woman is w.
- `wp[w][i] = m` if woman w's i^{th} favorite man is m.

`mp[1][0] = 3`
man 1 likes woman 3 the best

Men's Preference List

Man	0th	1st	2nd	3rd	4th
0	1	0	3	4	2
1	3	1	0	2	4
2	1	4	2	3	0
3	0	3	2	1	4
4	1	3	0	4	2

```
int mp[N][N];
int wp[N][N];
```


Initializing the Preference Lists

Could read from stdin.
We'll assign random lists for each man and woman.

- Use `randomPermutation` function from last time.
- Need N random permutations for men, and N for women.

```
int mp[N][N];
int wp[N][N];

for (m = 0; m < N; m++)
  randomPermutation(mp[m]);

for (w = 0; w < N; w++)
  randomPermutation(wp[w]);
```

`mp[m]` is man m's preference list array.

Dumping

Does woman w=2 prefer man m_1=3 to man m_2=0?

Women's Preference List

Woman	0th	1st	2nd	3rd	4th
0	4	0	1	3	2
1	2	1	3	0	4
2	1	2	3	4	0
3	0	4	3	2	1
4	3	1	4	2	0

search preference list sequentially
until m1 or m2 found

```
for (i = 0; i < N; i++) {
  if (wp[w][i] == m1) YES
  if (wp[w][i] == m2) NO
}
```

TOO SLOW if N is large, since need to repeat many times.

Keeping Track of Men's Proposals

Unmatched man proposes to most favorable woman to whom he hasn't already proposed.

How do we keep track of which woman a man has proposed to?

- Men propose in decreasing order of preference.
- Suffices to keep track of number of proposals in array.

  `propose[m] = i` if man m has proposed to i woman already.

initialize array

```
int props[N];
for (i = 0; i < N; i++)
  props[i] = 0;
```

find next woman to propose to

```
for (;;) {
  w = mp[m][props[m]];
  props[m]++;
  . . .
}
```

`props[m]` is next woman on preference list

make next proposal to woman `mp[m][props[m]]`

Try Out The Code

marriage.c

```
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#define N 500

int main(void) {
  int mp[N][N];   /* mp[m][i] = w if man m's ith favorite woman is w */
  int wp[N][N];   /* wp[w][i] = m if woman w's ith favorite man is m */
  int wife[N];    /* wife[m]  = w if m married to w                  */
  int husb[N];    /* husb[w]  = m if m married to w                  */
  int props[N];   /* props[m] = i if man m has proposed to i women   */

  int marriages = 0;   /* number of couples matched so far */
  int m, w;

  /* initialize men */
  for (m = 0; m < N; m++) {
    props[m] = 0;
    wife[m] = -1;
    randomPermutation(mp[m], N);
  }

  /* initialize women */
  for (w = 0; w < N; w++) {
    husb[w] = -1;
    randomPermutation(wp[w], N);
  }
```


Try Out The Code

marriage.c

```
  while (marriages < N) {
    /* find first unmatched man */
    for (m = 0; m < N; m++)
      if (wife[m] == -1) break;

    printf("man %d proposing:\n", m);
    /* propose to next women on list until successful */
    for (;;) {
      w = mp[m][props[m]];
      printf("  to woman %d", w);
      props[m]++;

      /* woman w unmatched */
      if (husb[w] == -1) {
        printf(" accepted\t(woman %d previously unmatched)\n", w);
        husb[w] = m; wife[m] = w;
        marriages++;
        break;
      }

      /* woman w prefers m to current mate */
      if (wr[w][m] < wr[w][husb[w]]) {
        printf(" accepted\t(woman %d dumps man %d)\n", w, husb[w]);
        wife[husb[w]] = -1;
        husb[w] = m; wife[m] = w;
        break;
      }
      /* otherwise m rejected by w */
      printf(" rejected\t(woman %d prefers %d)\n", w, husb[w]);
    }
  }
```


Try Out The Code

marriage.c

```
  printf("Stable matching\n");
  for (m = 0; m < N; m++)
    printf("%5d %5d\n", m, wife[m]);

  return 0;
}
```

Unix

```
% gcc marriage.c
% a.out

man 0 proposing:
  to woman 4 accepted    (woman 4 previously unmatched)
man 1 proposing:
  to woman 0 accepted    (woman 0 previously unmatched)
man 2 proposing:
  to woman 2 accepted    (woman 2 previously unmatched)
man 3 proposing:
  to woman 2 rejected    (woman 2 prefers 2)
  to woman 3 accepted    (woman 3 previously unmatched)
man 4 proposing:
  to woman 2 accepted    (woman 2 dumps man 2)
man 2 proposing:
  to woman 3 accepted    (woman 3 dumps man 3)
man 3 proposing:
  to woman 0 rejected    (woman 0 prefers 1)
  to woman 4 rejected    (woman 4 prefers 0)
  to woman 1 accepted    (woman 1 previously unmatched)

Stable matching
    0     4
    1     0
    2     3
    3     1
    4     2
```

Observation: code is REALLY slow for large N.

An Auxiliary Data Structure

Create a 2D array that stores men's ranking of women.

- `mr[m][w] = i` if man m's ranking of woman w is i.
- `wr[w][m] = i` if woman w's ranking of man m is i.

Men's Preference List

Man	0th	1st	2nd	3rd	4th
0	1	0	3	4	2
1	3	1	0	2	4
2	1	4	2	3	0
3	0	3	2	1	4
4	1	3	0	4	2

`mp[1][0] = 3`
man 1 likes woman 3 best

Men's Rankings

Man	0	1	2	3	4
0	1st	0th	4th	2nd	3rd
1	2nd	1st	3rd	0th	4th
2	4th	0th	2nd	3rd	1st
3	0th	3rd	2nd	1st	4th
4	2nd	0th	4th	1st	3rd

`mr[1][3] = 0`
man 1 likes woman 3 best

An Auxiliary Data Structure

Create a 2D array that stores men's ranking of women.

- `mr[m][w] = i` if man m's ranking of woman w is i.
- `wr[w][m] = i` if woman w's ranking of man m is i.

Does man m = 3 prefer woman w_1 = 2 to woman w_2 = 4?

```
if (mr[m][w1] < mr[m][w2])
  YES
else NO
```

Men's Rankings

Man	0	1	2	3	4
0	1st	0th	4th	2nd	3rd
1	2nd	1st	3rd	0th	4th
2	4th	0th	2nd	3rd	1st
3	0th	3rd	2nd	1st	4th
4	2nd	0th	4th	1st	3rd

`mr[m][w1] = 2`
`mr[m][w2] = 4`

Check if Marriage is Stable

Check if `husb[N]` and `wife[N]` correspond to a stable marriage.

- Good warmup and useful for debugging.
- Check every man-woman pair to see if they're unstable.
- Use ranking arrays.

isStable

```
int isStable(int husb[], int wife[],
             int mr[N][N], int wr[N][N]) {
  int m, w;
  for (m = 0; m < N; m++)
    for (w = 0; w < N; w++)
      if (mr[m][w] < mr[m][wife[m]]) &&
         (wr[w][m] < wr[w][husb[w]])
        return 0;
  return 1;
}
```

m prefers w to current wife

w prefers m to current husband

Check if Marriage is Stable

Check if `husb[N]` and `wife[N]` correspond to a stable marriage.

- Good warmup and useful for debugging.
- Check every man-woman pair to see if they're unstable.
- Use ranking arrays.

Time/space tradeoff for using auxiliary ranking arrays.

- Disadvantage: requires twice as much memory (storage).
- Advantage: dramatic speedup in running time (using 400 MHz Pentium II with N = 10,000).

Men vs. Women

Given input, there may be several stable marriages. Which one does algorithm find?

Fact 3. Propose-and-reject algorithm is MAN-OPTIMAL!
- Simultaneously best for each and every man.
- There is no stable marriage in which any single man individually does better.

Fact 4. Propose-and-reject algorithm is WOMAN-PESSIMAL.
- Simultaneously worst for each and every woman.
- There is no stable marriage in which any single woman individually does worse.

Fact 5. The man-optimal stable matching is weakly Pareto optimal.
- In every other matching (stable or unstable), at least one man does strictly worse.

Extensions

Yeah, but In real-world every woman is not willing to marry every man, and vice versa?
- Some participants declare others as “unacceptable” (prefer to be alone than with given partner).
- Algorithm extends to handle partial preference lists.

Also, there may be an unequal number of men and women.
- E.g., 150 men, 100 women.
- Algorithm extends.

What about limited polygamy?
- E.g., Bill wants 3 women.
- Algorithm extends.

Application

Matching medical school residents to hospitals. (NRMP)
- Hospitals ~ Men (limited polygamy allowed).
- Residents ~ Women.
- Original use just after WWII (predates computer usage).
- Ides of March, 13,000+ residents.

Rural hospital dilemma.
- Certain hospitals (mainly in rural areas) were unpopular and declared unacceptable by many residents.
- Rural hospitals were under-subscribed in NRMP matching.
- How can we find stable matching that benefits “rural hospitals”?

Rural Hospital Theorem:

Deceit: Machiavelli Meets Gale-Shapley

Is there any incentive for a participant to misrepresent his/her preferences?
- Assume you know men’s propose-and-reject algorithm will be run.
- Assume that you know the preference lists of all other participants.

Fact 6.

Deceit: Machiavelli Meets Gale-Shapley

Is there any incentive for a participant to misrepresent his/her preferences?

Fact 7.

Men's Preferences

	0^{th}	1^{st}	2^{nd}
Xavier	A	B	C
Yancy	B	A	C
Zeus	A	B	C

Women's Preferences

	0^{th}	1^{st}	2^{nd}
Amy	Y	X	Z
Bertha	X	Y	Z
Clare	X	Y	Z

Amy lies.

	0^{th}	1^{st}	2^{nd}
Amy	Y	**Z**	**X**
Bertha	X	Y	Z
Clare	X	Y	Z

Lessons Learned

Powerful ideas learned in COS 126.

- Combine to obtain neat and useful algorithms.