Objectives

- Review: Asymptotic running times
- Implementing Gale-Shapley algorithm
- Classes of running times

Review Asymptotic Bounds

- What does O(f(n)) mean?
- What are the other two bounds we discussed?
- We discussed three classes of running times
  - What are they?
  - Order them by their growth rates

Review: Our Process

1. Understand/identify problem
   - Simplify as appropriate
2. Design a solution
3. Analyze
   - Correctness, efficiency
   - May need to go back to step 2 and try again
4. Implement
   - Within bounds shown in analysis

IMPLEMENTING GALE-SHAPLEY ALGORITHM

Review: Gale-Shapley Stable Matching Algorithm

```
Initialize each person to be free
while (some man is free and hasn't proposed to every woman)
    Choose such a man m
    w = 1st woman on m's list to whom m has not yet proposed
    if (w is free)
        assign m and w to be engaged
    else if (w prefers m to her fiancé m')
        assign m and w to be engaged and m' to be free
    else
        w rejects m
```


How Can We Implement The Algorithm Efficiently?

- What is our goal for the implementation's runtime?
- What do we need to model?
- How should we represent them?

How Can We Implement The Algorithm Efficiently?

- What is our goal for the implementation's runtime?
  - $O(N^2)$
- What do we need to model?
- How should we represent them?

Stable Matching Implementation

- What do we need to represent?
- How should we represent them?

Data	How represented
Men, Women	
Preference lists	
Unmatched men	
Who men proposed to	
Engagements	

What's the difference between an array and a list?

Arrays

- *Fixed* number of elements
- What is the runtime of
  - Determining the value of the i^{th} item in the array?
  - Determining if a value *e* is in the array?
  - Determining if a value *e* is in the array if the array is sorted?

Array Operations' Running Times

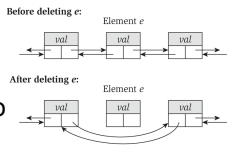
Operation	Running Time
Value of i^{th} item	O(1) → direct access
If *e* is in the array	O(n) → look through all the elements
If *e* is in the array if sorted	O(log n) → binary search

Limitation of arrays?

Fixed size, so difficult to add/delete elements

Lists

- Dynamic set of elements
  - Linked list
  - Doubly linked list
- What is the running time to
  - Add an element to the list?
  - Delete an element from the list?
  - Find an element *e* in the list?
  - Find the i^{th} element in the list?



List Operations' Running Time

Operation	Running Time
Add element	O(1)
Delete element	O(1)
Find element	O(n)
Find i^{th} element	O(i)

Disadvantage of list instead of array?

Finding i^{th} element is slower

Converting between Lists and Arrays (and Vice Versa)

- What is the running time of converting a list to an array?
- An array to a list?

O(n)

Stable Matching Implementation

- What do we need to represent? How should we represent them?

Data	How represented
Men, Women	
Preference lists	
Unmatched men	
Who men proposed to	
Engagements	

Stable Matching Implementation

- What do we need to represent? How should we represent them?

Data	How represented
Men, Women	Integers (like ids)
Preference lists	Array of arrays (2d array)
Unmatched men	List
Who men proposed to	Integer for each man → Array of integers
Engagements	2 Arrays

Asymptotic Analysis of Gale-Shapley Alg

```
Initialize each person to be free
while (some man is free and hasn't proposed to every woman)
    Choose such a man m
    w = 1st woman on m's list to whom m has not yet proposed
    if (w is free)
        assign m and w to be engaged
    else if (w prefers m to her fiancé m')
        assign m and w to be engaged and m' to be free
    else
        w rejects m
```

What is the running time of each part of the algorithm?
What is the total running time of the algorithm?

Efficient Implementation

- Women rejecting/accepting: determine does woman `w` prefer man `m` to man `m'`?
  - For each woman, create array of men with her preference
    - *inverse* of preference list
  - Constant time access for each query after O(n) preprocessing

Amy	1st	2nd	3rd	4th	5th	6th	7th	8th
Pref	8	3	7	1	4	5	6	2

Contains man's id

For each man, how does he rank?

Amy	1	2	3	4	5	6	7	8
Inverse	4th	8th	2nd	5th	6th	7th	3rd	1st

Amy prefers man 3 to 6 since `inverse[3] < inverse[6]`

2 7

```
for i = 1 to n
    inverse[ pref[i] ] = i
```


Asymptotic Analysis of Gale-Shapley Alg

Not explicitly in the algorithm, but we need to make the inverse array before the while loop too.

```
Initialize each person to be free      O(n²)
while (some man is free and hasn't proposed to every woman)   O(n²)
    Choose such a man m                O(1)        O(1)
    w = 1st woman on m's list to whom m has not yet proposed
    if (w is free)                     O(1)
        assign m and w to be engaged   O(1)
    else if (w prefers m to her fiancé m')   O(1)   Using inverse array
        assign m and w to be engaged and m' to be free   O(1)
    else
        w rejects m                    O(1)
```

Total: $O(n^2)$

Continuing from Friday…

A SURVEY OF COMMON RUNNING TIMES

Cubic Time: $O(n^3)$

- Examples?

Cubic Time: $O(n^3)$

- Enumerate all triples of elements

Cubic Time: $O(n^3)$

- Set disjointness. Given n sets S_1, …, S_n each of which is a subset of 1, 2, …, n, is there some pair of these which are disjoint?
- $O(n^3)$ solution. For each pair of sets, determine if they are disjoint

```
foreach set Si
   foreach other set Sj
      foreach element p of Si
         determine whether p also belongs to Sj

      if (no element of Si belongs to Sj)
         report that Si and Sj are disjoint
```


Polynomial Time: $O(n^k)$ Time

- To get all pairs, the algorithm is $O(n^2)$
- To get all 3-tuples, the algorithm is $O(n^3)$

What is an example of an $O(n^k)$ algorithm?

All subsets of size k

Polynomial Time: $O(n^k)$ Time

- Independent set of size k. Given a graph, are there k nodes such that no two are joined by an edge?
  - k is a constant

Assignments

- Review Chapter 2
  - Finishing up on Wednesday
- Journal due Tuesday at 11:59:59
- Problem Set 1 due Friday in class
  - FAQ of commonly asked questions on course web page