P vs NP

CHOTCHKIES RESTAURANT

APPETIZERS

Mixed Fruit 2.15
French Fries 2.75
Side Salad 3.35
Hot Wings 3.55
Mozzarella Sticks 4.20
Sampler Plate 5.80

SANDWICHES

Barbecue 6.55

WE'D LIKE EXACTLY $15.05 WORTH OF APPETIZERS, PLEASE.

...EXACTLY? UHH...

HERE, THESE PAPERS ON THE KNAPSACK PROBLEM MIGHT HELP YOU OUT.

LISTEN, I HAVE SIX other TABLES TO GET TO —

— AS FAST as possible, of COURSE. WANT SOMETHING on TRAVELING SALESMAN?
Projects

```java
while(noSuccess)
{
    tryAgain();
    if(Dead)
        break;
}
```
So far in this class, we have only really considered algorithms bound by polynomial time (thus “P”)

Sorting = $O(n)$ or $O(n \log n) < O(n^2)$
Sorting = $O(n+k) < O(n)$
Shortest path = $O(n^2)$
All-pairs shortest path = $O(n^3)$
Some problems cannot be solved in polynomial time (not poly = “NP”)

These tend to have an exponential growth, $O(2^n)$

Typically we have to settle for approximate solutions
One famous problem is the traveling salesperson problem:
P vs NP

We can generalize a relationship between P and NP problems:

If you can *solve* an NP problem...

... you can *verify* it in polynomial time
Oddly enough, we cannot tell if:
NP is harder than P... or the same

If they are the same, this would have vast implications for all areas

It means you could reduce solving to checking
P vs NP

Is tasking good food as easy as being an good chef?

Is grading an exam as easy as taking an exam?

Is knowing good sports plays as easy as performing them?
P vs NP

There are actually 3 different (main) levels of NP problems:

NP - not polynomial
NP hard - if you can solve this, you can solve any NP problem
NP complete - both NP and NP-hard (some NP hard outside of NP)
P vs NP
P vs NP

P ≠ NP

P = NP = NP-Complete

NP-Hard

NP-Complete

NP
How to tell P vs NP

The main way we define these are using reduction mappings:

This is basically showing: if I could solve A, I could solve B efficiently

This means A is a “harder” problem than “B”
How to tell P vs NP

Examples:
Sorting is harder than min (selection)

Sort, then take first element
$O(A) + O(1) \geq O(B)$

The key is the conversion does not take non-polynomial time
Projects:
- Required section (in outline)
- Citing references (small tutorial)
  (loose format)
  (not too deep)
  (but MUST do)