GOAL:
- Introduce basic concepts in Complexity Theory.

PLAN:
- Meet Celebrities and Computations
- Growth Rate and Tractability
- Reducibility
- ... etc. ...
Drama At the Oscars

Problem:

- seat all guests around a table, so people who sit next to each other get along.
# How Can a Catastrophe be Avoided?

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>

- **Hearts**: Represent love or positive outcomes.
- **Lightning Bolts**: Represent negative outcomes or catastrophes.

Each cell in the table represents a situation where hearts and lightning bolts are distributed to illustrate the concept of avoiding catastrophes in relationships or scenarios.
Getting It Right
Observation:

- Given a seating one can efficiently check if all guests get along with their neighbors

For each seating arrangement:
Check if all guests are OK with neighbors
Stop if a good arrangement is found

How much time would it take? (worse case)
For each seating arrangement:
Check if all guests are OK with neighbors
Stop if a good arrangement is found

How much time would it take? (worse case)

<table>
<thead>
<tr>
<th>Guests</th>
<th>Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>(N-1)!</td>
</tr>
<tr>
<td>5</td>
<td>24</td>
</tr>
<tr>
<td>15</td>
<td>87178290</td>
</tr>
<tr>
<td>100</td>
<td>$\approx 9 \cdot 10^{135}$</td>
</tr>
</tbody>
</table>

say our computer is capable of $10^{10}$ instructions per second, this will still take $\approx 3 \cdot 10^{138}$ years!

Can you do better?
• Plan a trip that visits every location exactly once.
For each site

Try out all reachable sites not yet visited

Backtrack and retry

Repeat the process until stuck

Naive Algorithm (Backtracking)
### How Much Time?

<table>
<thead>
<tr>
<th>Sites</th>
<th>Steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N!</td>
</tr>
<tr>
<td>5</td>
<td>120</td>
</tr>
<tr>
<td>15</td>
<td>1307674368000</td>
</tr>
<tr>
<td>100</td>
<td>≈9·10^{157}</td>
</tr>
</tbody>
</table>

On a computer that can check 10,000 options per second, this will still take 4 years!
Is a Problem Tractable?

- and here’s an efficient algorithm that solves it

- and I can prove it

- and if neither is the case?
Growth Rate: rough classification

\[ n! = 2^{O(n \log n)} \]
Basic split in time-complexity

Maybe reasonable

polynomial
\equiv n^{O(1)}

exponential
\equiv 2^{n^{O(1)}}

 Totally unreasonable
If

- assuming an efficient procedure for $B$
  there is an efficient procedure for $A$

Then

- $A$ cannot be radically harder than $B$
In other words:

- **A** cannot be radically harder than **B**

**B** is at least as hard as **A**
Find someone who can seat next to everyone
Reduce Tour to Seating

Completeness:

- If there's a tour, there's a way to seat all the guests around the table.

Soundness:

- If there's a seating, we can easily find a tour path (no tour, no seating).

QED

- Seating is at least as hard as tour
• to show a very strong correlation between their complexity
• prove they are intractable
• find an efficient algorithm for problems

So Far
Interestingly, we can also reduce the seating problem to the tour problem.

Furthermore, there is a whole class of problems, which can be pair-wise efficiently reduced to each other.
NP and P

• Efficiently computable

P

• Solution efficiently verifiable

Is $P=NP$?
NP

NPC

Contains thousands of distinct problems

Each reducible to all others

Exponential-time algorithms

Efficient algorithms
How can Complexity make you a Millionaire?

The “P vs. NP” question is the most fundamental of CS

Resolving it would bring you great honor...

... as well as significant fortune... www.claymath.org/

Philosophically: if $P=NP$

• Human ingenuity is redundant!

• So would mathematicians be!!

Is nature nondeterministic?
What's Ahead?

• we'll review basic questions explored through the course.
Generalized Tour Problem

- Each segment of the tour problem now has a **cost**
- find a **least-costly** tour
Is Running Time the only Resource?

- What about memory (space)?
- Any other?
Players take turns choosing a word whose first letter matches the other player’s last word.
Can one compute a winning strategy?
How much time would it take?
How much space?
We have introduced two problems:
1. Seating $=$ HAMILTONIAN-CYCLE
2. Tour $=$ HAMILTONIAN-PATH

Unable to settle their complexity we, nevertheless, showed strong correlations between them.

These problems are representatives of a large class of problems: NPC.
• Approximation
• Space-bounded computations
<table>
<thead>
<tr>
<th>Complexity Theory</th>
<th>Computations</th>
<th>Completeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hamiltonian Path</td>
<td>Growth Rate</td>
<td>Completeness</td>
</tr>
<tr>
<td>Reducibility</td>
<td></td>
<td>Soundness</td>
</tr>
<tr>
<td>Complexity Classes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NPC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exponential Time</td>
<td><a href="http://www.claymath.org">www.claymath.org</a></td>
<td>Approximation</td>
</tr>
</tbody>
</table>