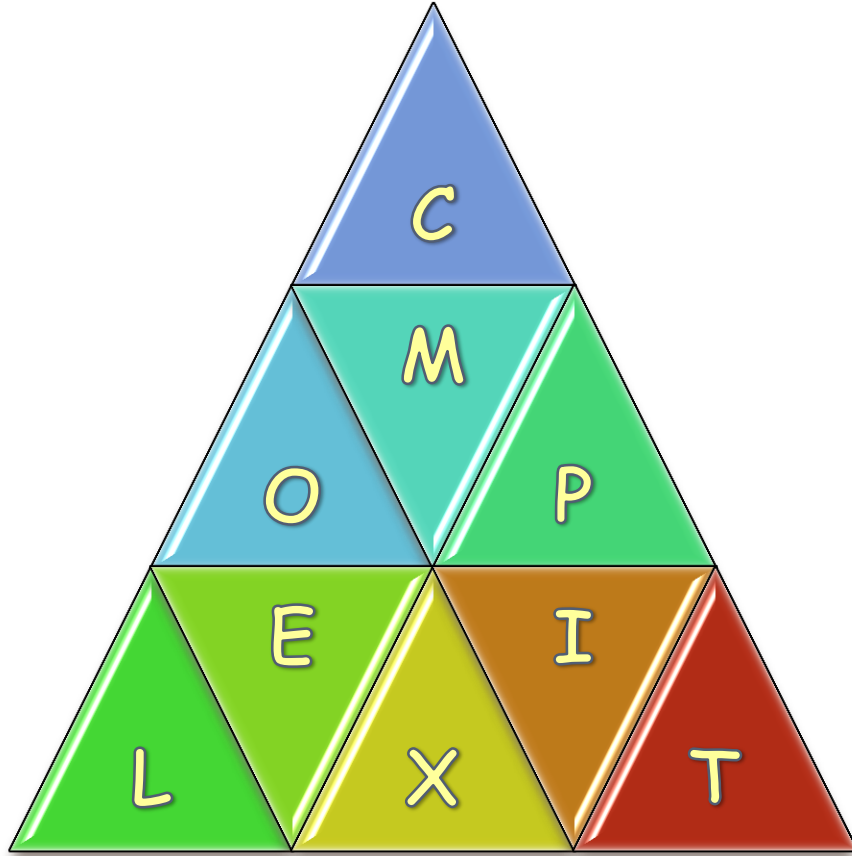




Goal:

- Introduce basic concepts in Complexity Theory.

Plan:

- Meet Celebrities and Computations
- Growth Rate and Tractability
- Reducibility
- ... etc. ...

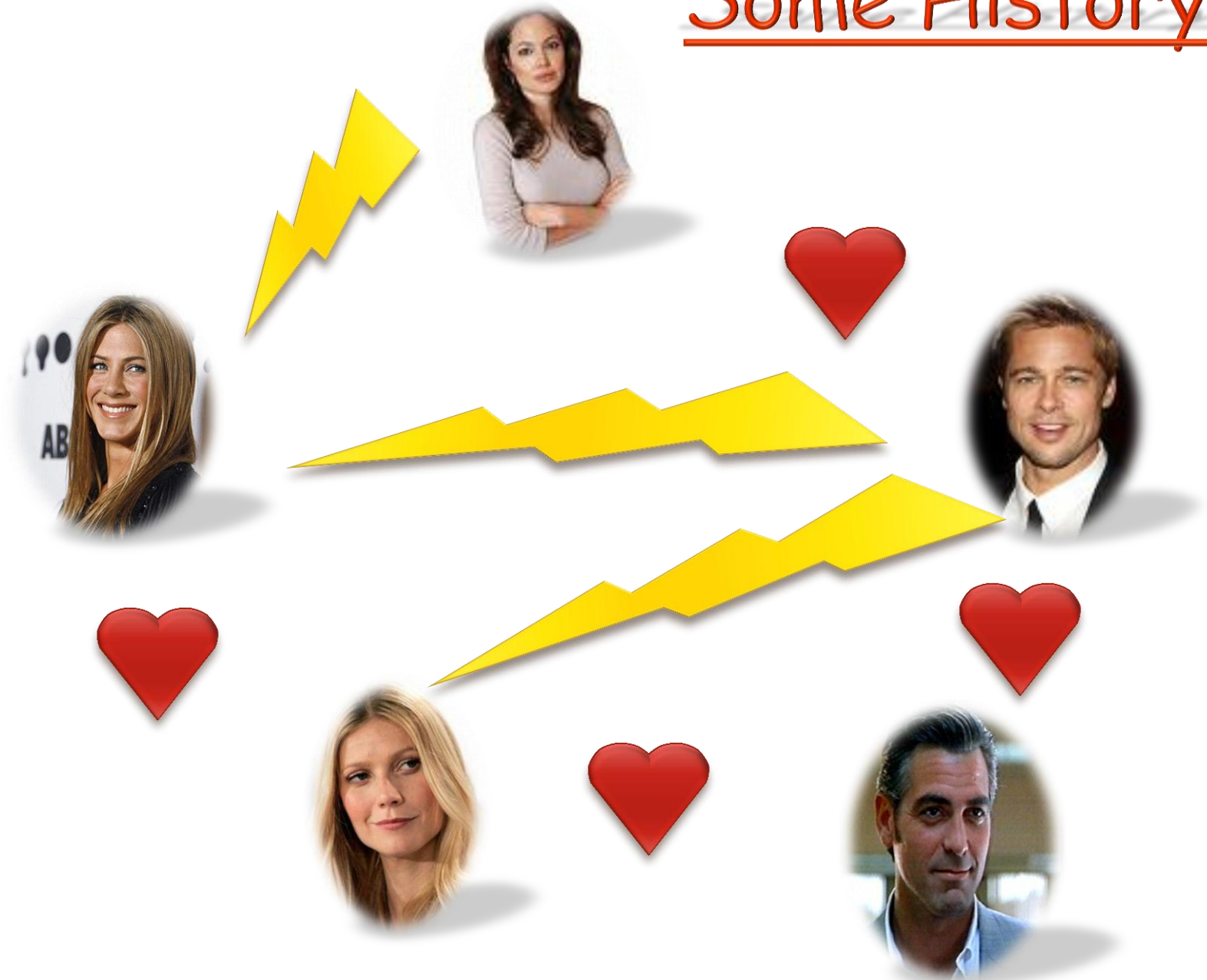
Drama At the Oscars

Problem:

- seat all guests around a table, so people who sit next to each other get along.



Some History...



How Can a Catastrophe be Avoided?



Getting It Right



Naive Algorithm

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*)

Naive Algorithm

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)

Guests	Steps
N	$(N-1)!$
5	24
15	871782912
100	$\approx 9 \cdot 10^{155}$

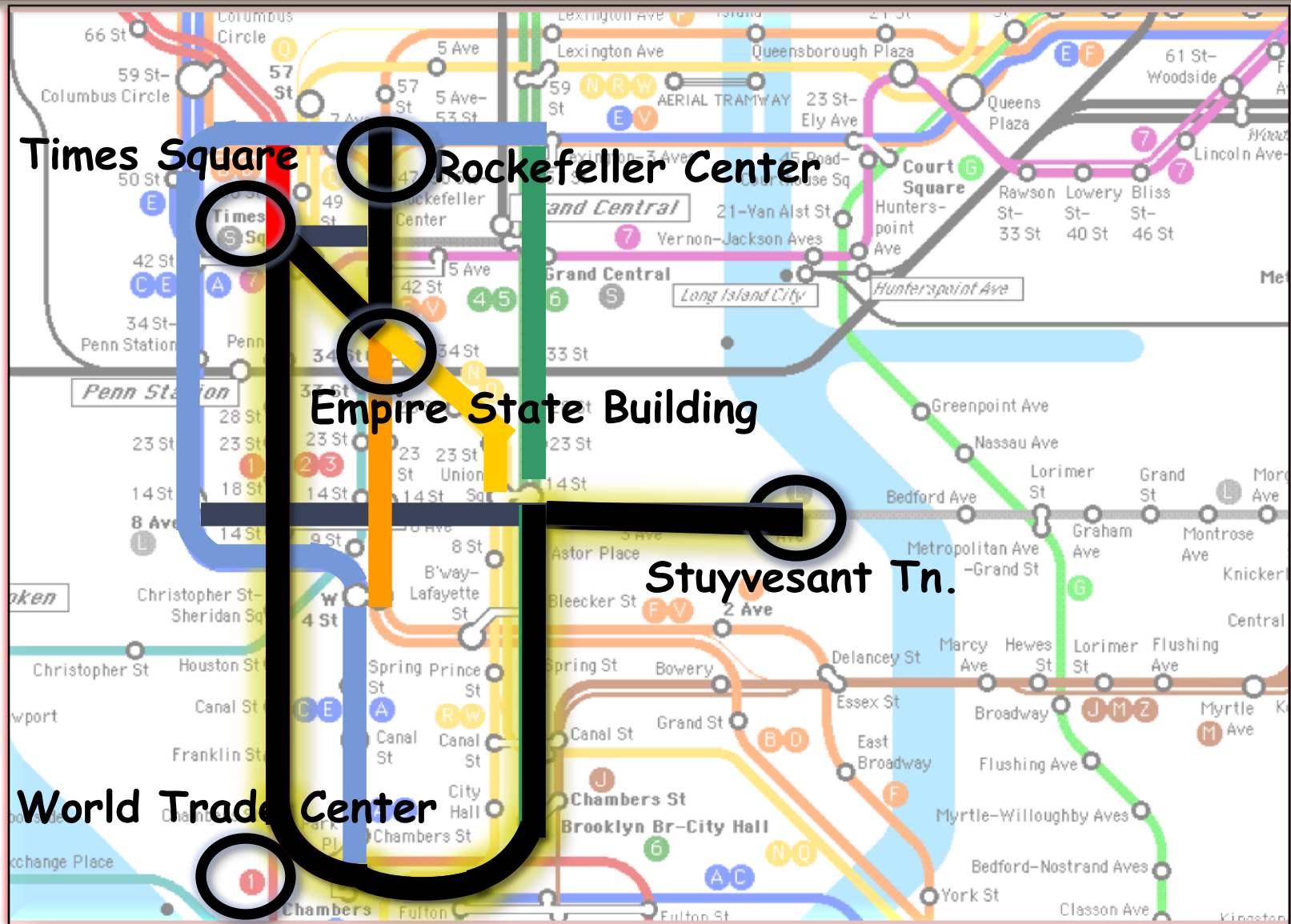
• say our computer is capable of 10^{10} instructions per second, this will still take $\approx 3 \cdot 10^{138}$ years!



Can you do better? ⁸

Tour Problem

- Plan a trip that visits every location exactly once.



Naive Algorithm (Backtracking)

For each site

Try out all
reachable sites
not yet visited

Backtrack
and retry

Repeat the
process until
stuck

How Much Time?

Sites	Steps
N	$N!$
5	120
15	1307674368000
100	$\approx 9 \cdot 10^{157}$

- On a computer that can check **10,000** options per second, this still takes **4 years!**

Is a Problem Tractable?



- and here's an **efficient algorithm** that solves it



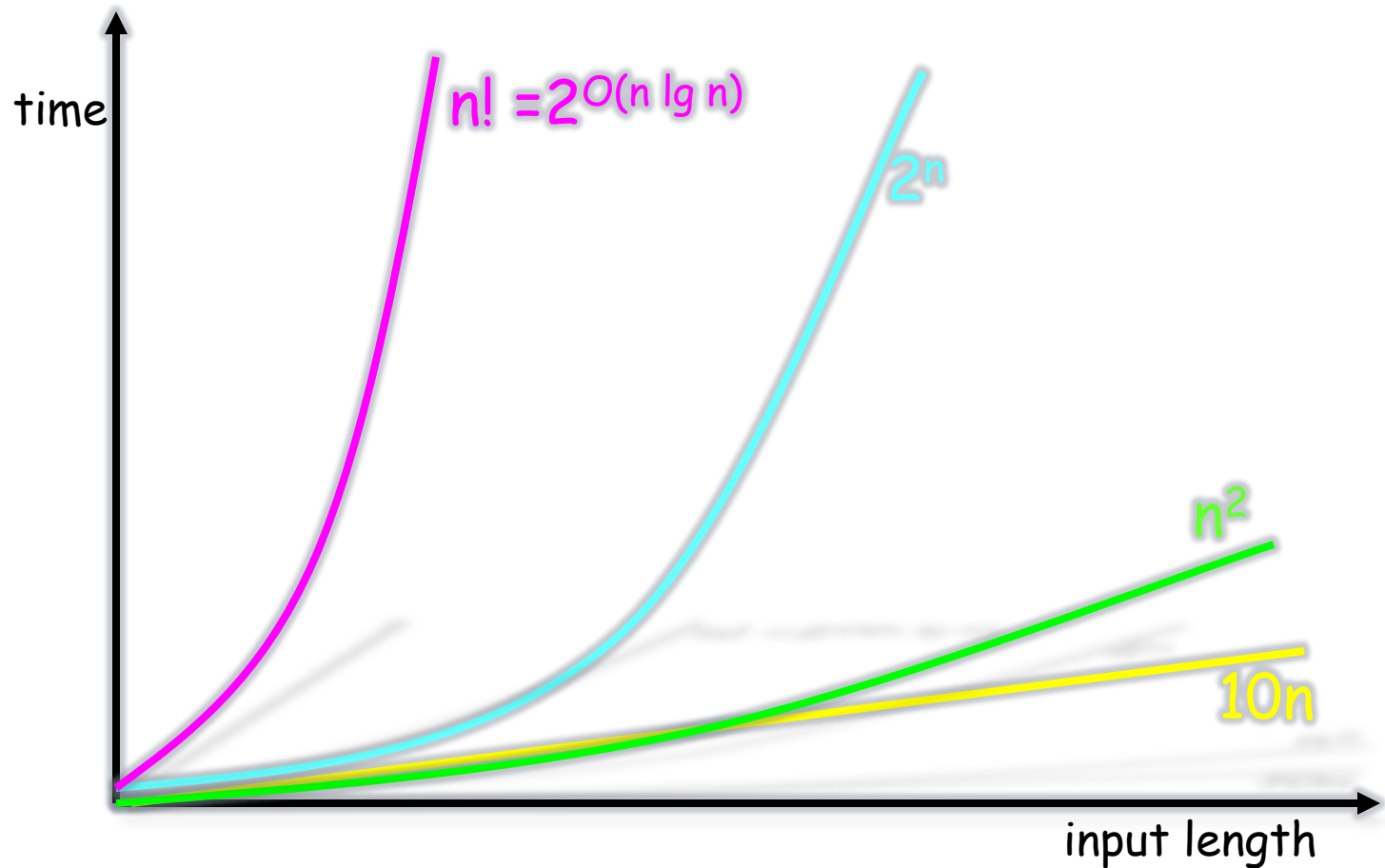
- and *I can prove it*



- and if **neither** is the case?



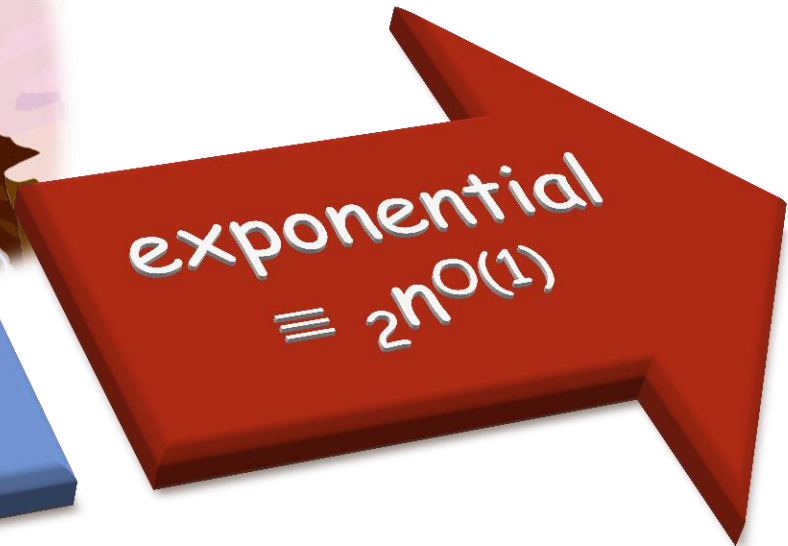
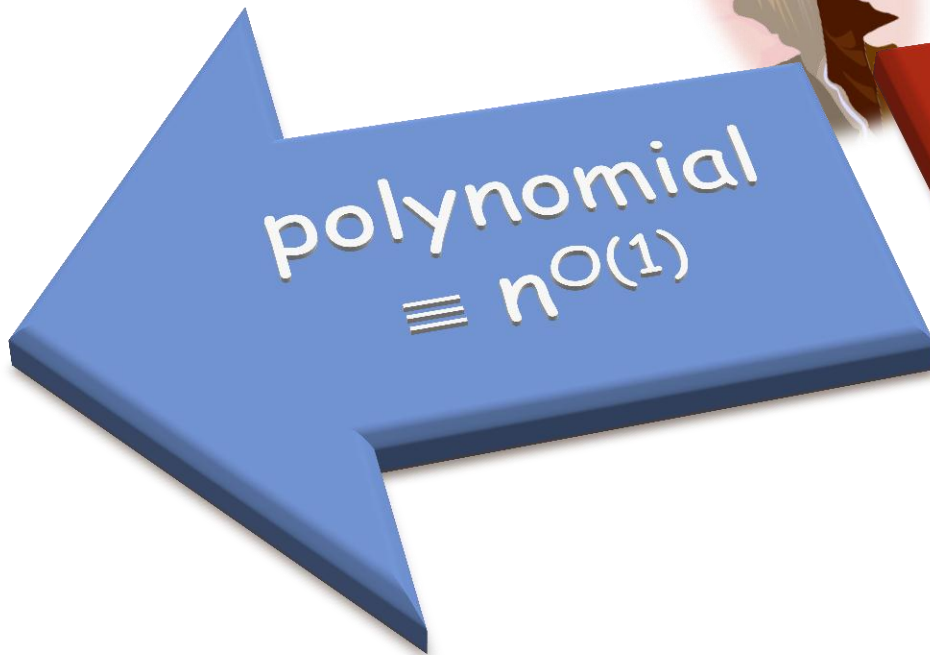
Growth Rate: rough classification



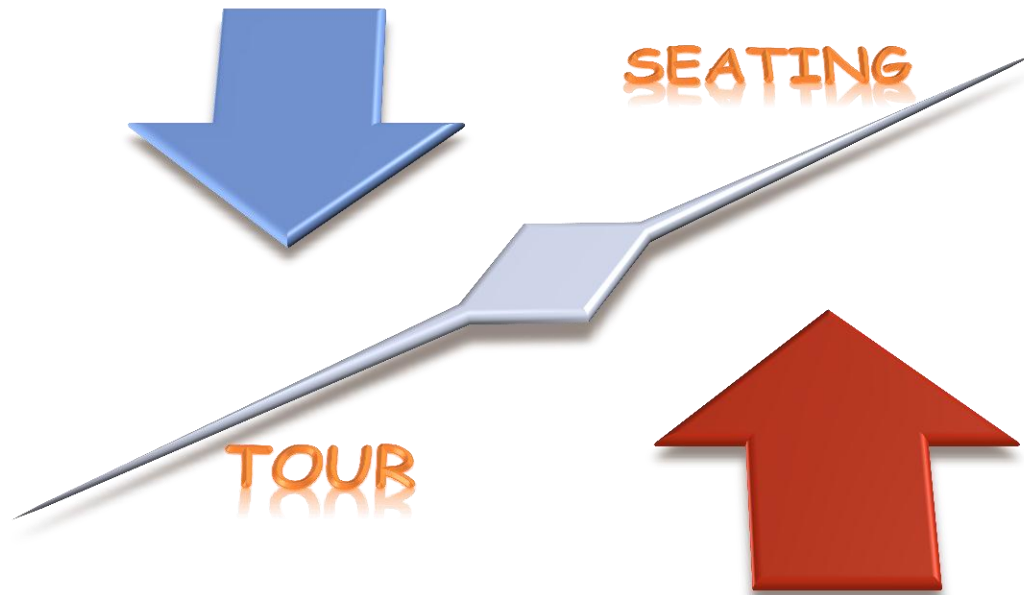
Basic split in time-complexity

Maybe
reasonable

Totally
unreasonable



Which is Harder?



Relations Between Problems

If

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

an efficient procedure for A
using

an efficient procedure for
 B

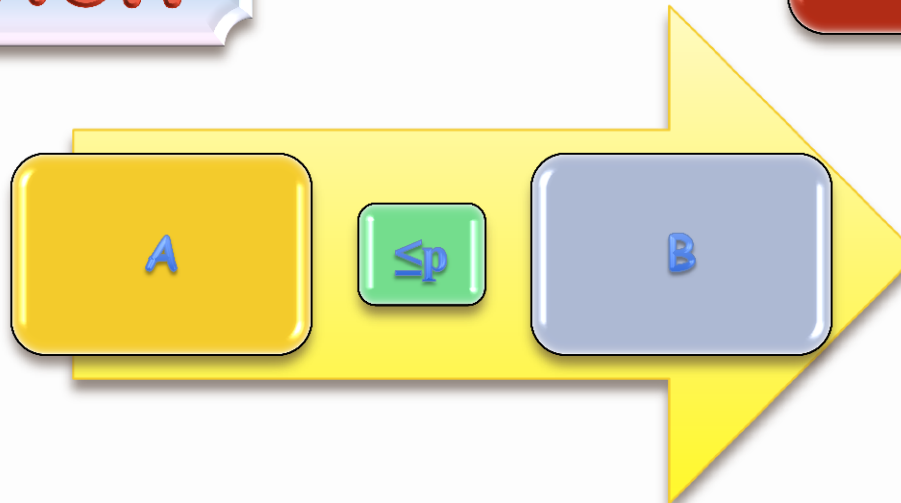
Then

- A cannot be radically harder than B

an efficient procedure for A
using

an efficient procedure for
 B

Notation



Reductions

A cannot be radically
harder than B

• In other words:

B is at least as hard
as A

Reduce Tour to Seating

					
		✓	✗	✓	✗
	✓		✓	✗	✗
	✗	✓		✓	✓
	✓	✗	✓		✓
	✗	✗	✓	✓	

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

So Far

COULD
NOT:

- find an **efficient algorithm** for problems



NOR

- prove they are ***intractable***



DID
MANAGE:

- to show a very **strong correlation** between ***their complexity***





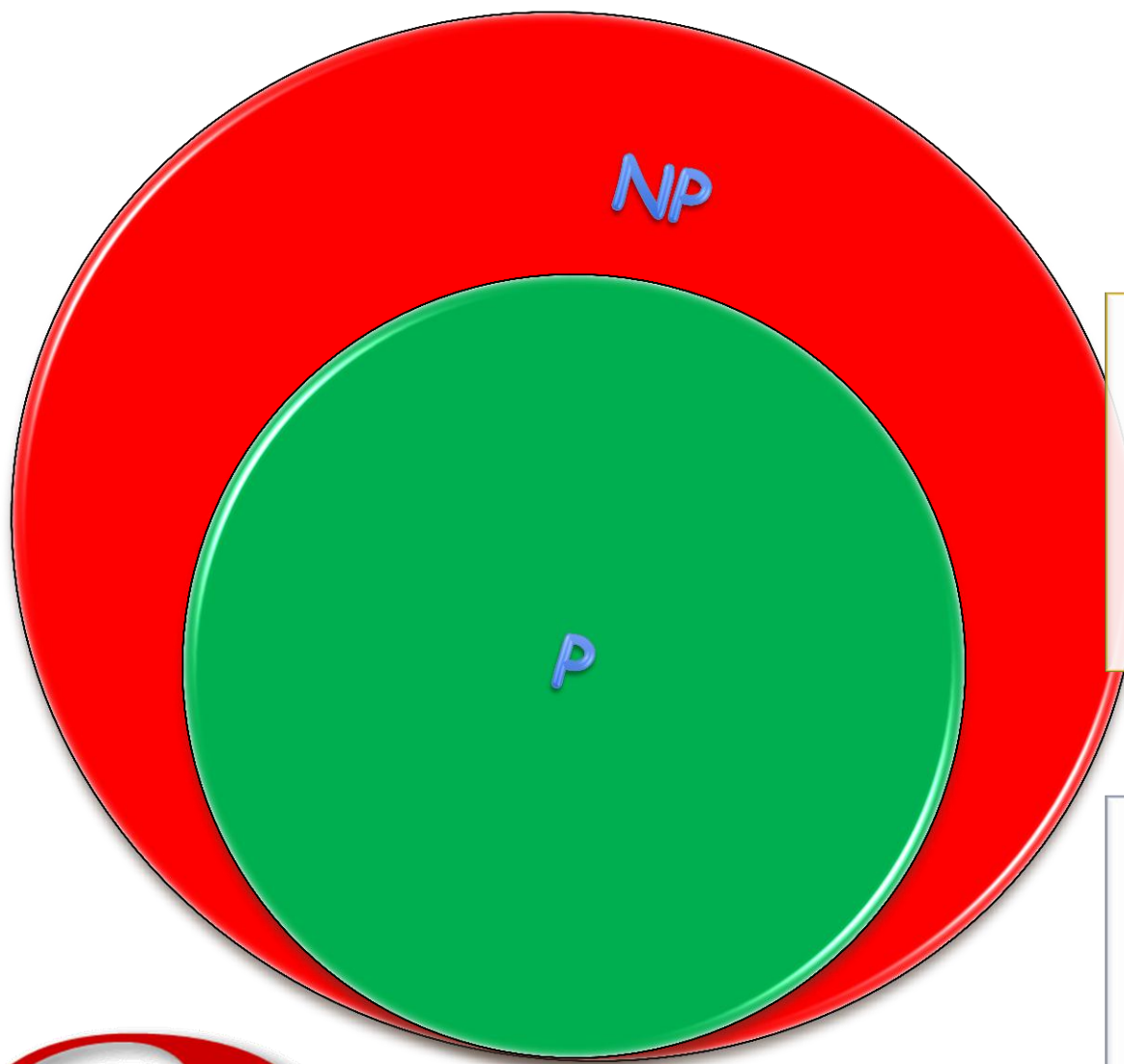
Interestingly,
we can also reduce the
seating problem to the
tour problem.

Can
you?



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

NP and P



- Efficiently computable



- Solution efficiently verifiable

Does
P=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?

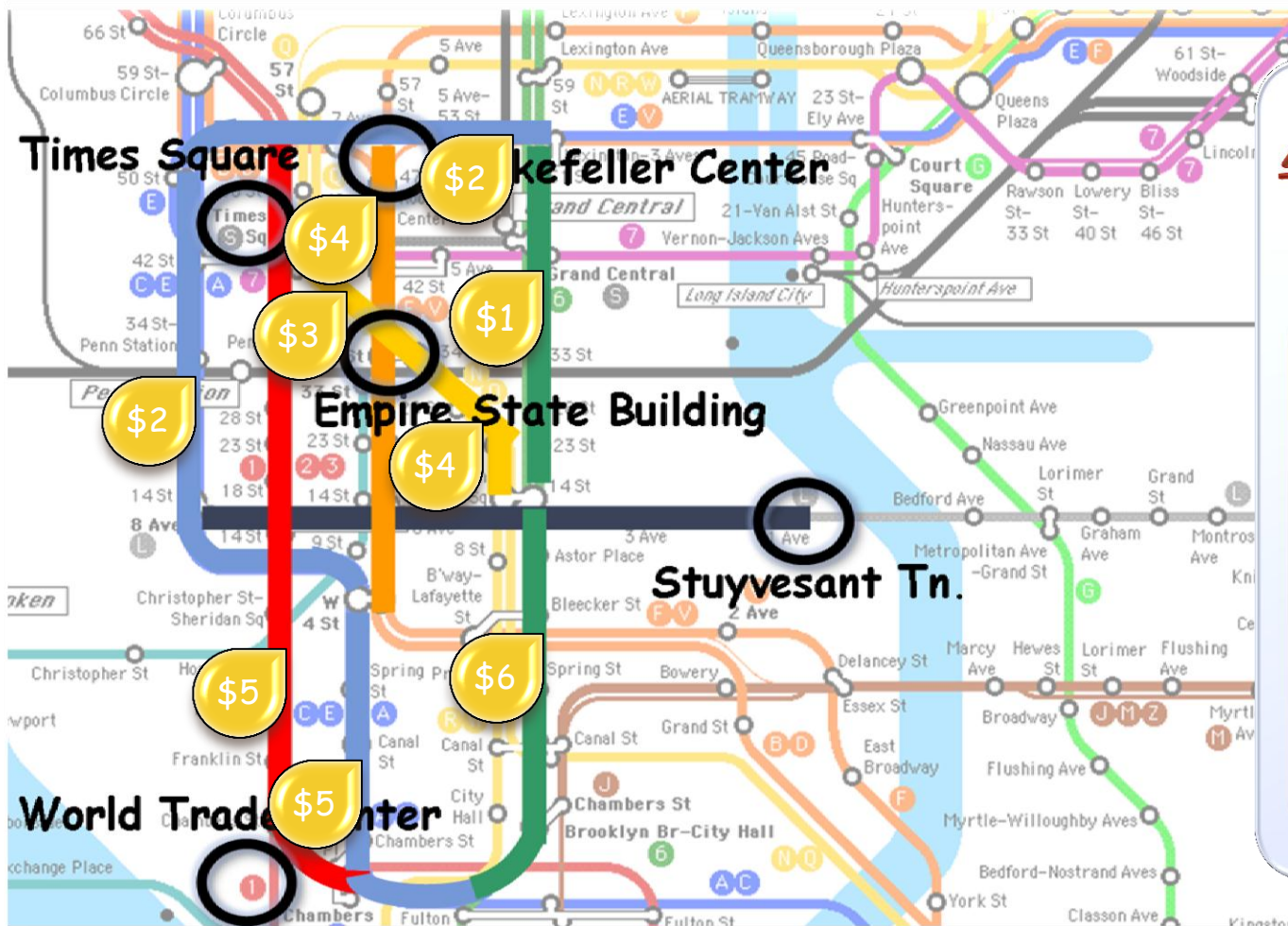


Next:

- 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



Approximate the optimal tour?

i.e. - find a tour that costs, say, no more than twice as much as the least costly.



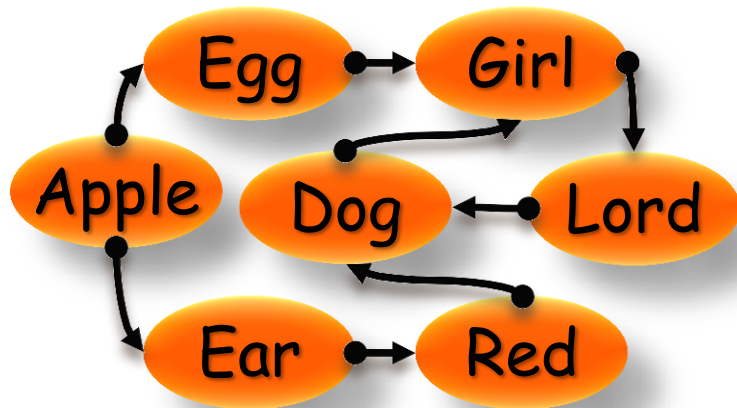
Is Running
Time the
only
Resource?

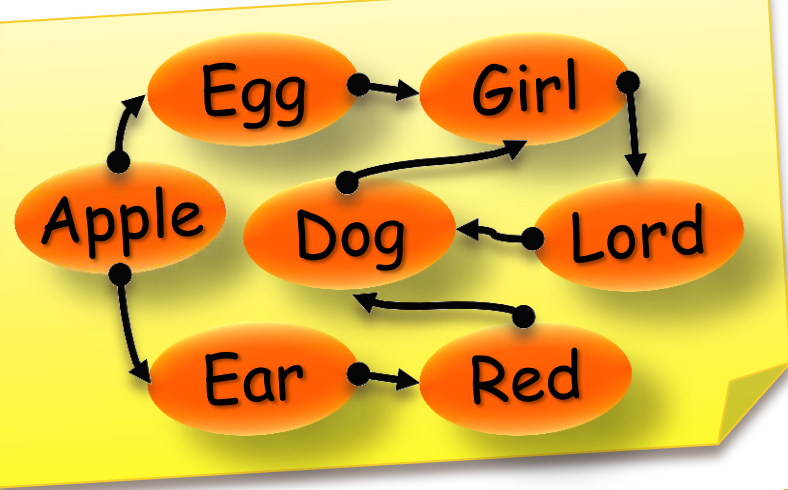
- What about memory (*space*)?
- Any other?

Games

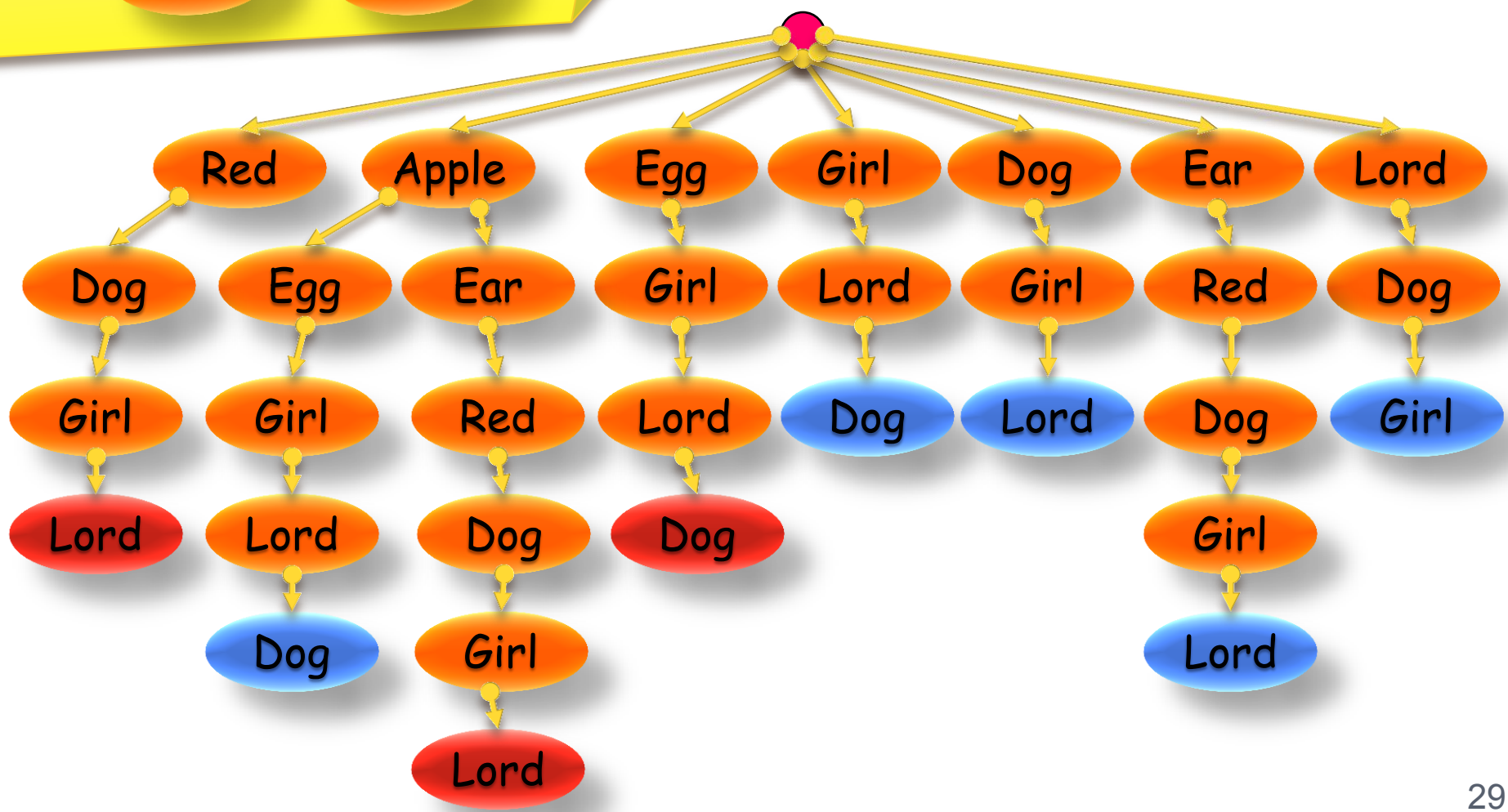
Word Games:

Players take turns
choose a word whose
first letter matches
other player's last





- Can one compute a winning strategy?
- How much time would it take?
- How much space?



Summary

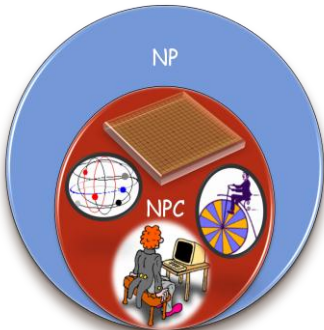


We have introduced two problems:

1. **Seating** $\equiv$ HAMILTONIAN-CYCLE
2. **Tour** $\equiv$ 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

Complexity
Theory

Computations

Completeness



Hamiltonian
Path

Growth Rate

Completeness



Reducibility

Soundness

Complexity
Classes

P

NP

NPC

Exponential
Time

www.claymath.org

Approximation

WWindex