

CSCI 361 Lecture 22:  
Complexity Theory Wrap Up

Shikha Singh

# Announcements & Logistics

- Hand in **1-page paper draft**
- **Survey paper deadlines:**
  - 10 min presentation + Q&A on Dec 5
  - Final paper due Dec 6 on Gradescope
- End early today to allow time for student
- **Self-scheduled final exam** between **Dec 7-15**
  - 2.5 hr exam
  - Similar format as midterm
  - In-class review session on Tuesday Dec 3

# Last Time

- Discussed survey paper topics
- Proved Cook-Levin Theorem
  - SAT is NP complete
  - Discussed other NP complete problems

# Today

- Discuss whether LLMs can solve all our problems
- Wrap up complexity theory

# List of NPC Problems

Have not shown all but similar reductions

- **Satisfiability:** SAT/ 3-SAT
- INDEPENDENT SET and CLIQUE
- **Covering problems:** VERTEX COVER, SET COVER
- **Coloring problem:** 3-COLOR
- **Sequencing problems:**
  - Traveling salesman problem
  - Hamiltonian cycle / path
- **Packing problems:** Subset-Sum, Knapsack (CSCI 256)

# Many More hard computational problems

**Aerospace engineering.** Optimal mesh partitioning for finite elements.

**Biology.** Phylogeny reconstruction.

**Chemical engineering.** Heat exchanger network synthesis.

**Chemistry.** Protein folding.

**Civil engineering.** Equilibrium of urban traffic flow.

**Economics.** Computation of arbitrage in financial markets with friction.

**Electrical engineering.** VLSI layout.

**Environmental engineering.** Optimal placement of contaminant sensors.

**Financial engineering.** Minimum risk portfolio of given return.

**Game theory.** Nash equilibrium that maximizes social welfare.

**Mathematics.** Given integer  $a_1, \dots, a_n$ , compute

**Mechanical engineering.** Structure of turbulence in sheared flows.

**Medicine.** Reconstructing 3d shape from biplane angiocardiogram.

**Operations research.** Traveling salesperson problem.

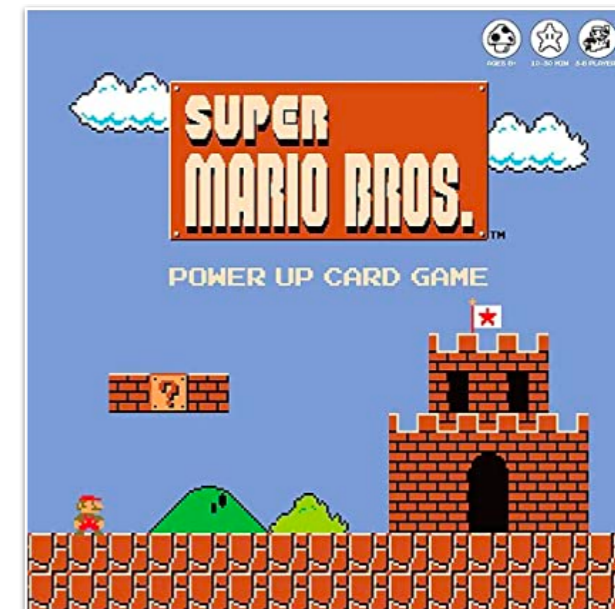
**Physics.** Partition function of 3d Ising model.

**Politics.** Shapley–Shubik voting power.

**Recreation.** Versions of Sudoku, Checkers, Minesweeper, Tetris, Rubik's Cube.

# Fun NP-hard Games

- **MINESWEEPER** (from CIRCUIT-SAT)
- **SODUKO** (from 3-SAT)
- **TETRIS** (from 3PARTITION)
- **SOLITAIRE** (from 3PARTITION)
- **SUPER MARIO BROTHERS** (from 3-SAT)
- **CANDY CRUSH SAGA** (from 3-SAT variant)
- **PAC-MAN** (from Hamiltonian Cycle)
- **RUBIC's CUBE** (recent 2017 result, from Hamiltonian Cycle)
- **TRAINYARD** (from Dominating Set)



# NP Completeness: Are We Doomed?

- A lot of optimization problems are NP-complete
  - Scheduling problems
  - Packing problems for resource allocation
- Most of these get solved routinely
  - Why does this not contradict their NP complete status?
- Heuristics vs algorithms with worst-case guarantees



# Solving NP Hard Problem: Pathways

- Algorithms with provable guarantees
  - Approximation algorithms
  - Parameterized algorithms
  - Randomized algorithms
- Heuristics without running time guarantees

## [Discussion]

Can ChatGPT Solve all Our Problems?

# What Problems can LLMs Solve?

- New research direction: understanding how to characterize the "computational power" of transformer models
- Our current understand:
  - Limited complexity in terms of solvability: contained within class  $\mathbf{TC}_0$  (<https://arxiv.org/abs/2310.07923>, Merrill and Sabharwal '23])
    - Example problems in class: sorting  $n$ -bit numbers, multiplying two  $n$ -bit numbers, etc
  - What graph problems can transformers solve? <https://arxiv.org/pdf/2405.18512> [Sanford et al. 2024]

# NPHardEval: Dynamic Benchmark on Reasoning Ability of Large Language Models via Complexity Classes

Lizhou Fan<sup>1\*</sup> Wenyue Hua<sup>2\*</sup> Lingyao Li<sup>1</sup> Haoyang Ling<sup>1</sup> Yongfeng Zhang<sup>2</sup>

<sup>1</sup>School of Information, University of Michigan, Ann Arbor, MI 48103

<sup>2</sup>Department of Computer Science, Rutgers University, New Brunswick, NJ 08854  
{lizhouf, lingyaol, hyfrankl}@umich.edu, {wenyue.hua, yongfeng.zhang}@rutgers.edu

\*Lizhou Fan and Wenyue Hua contribute equally.

## Abstract

Complex reasoning ability is one of the most important features of Large Language Models (LLMs). Numerous benchmarks have been established to assess the reasoning abilities of LLMs. However, they are inadequate in offering a rigorous evaluation and prone to the risk of overfitting and memorization, as these publicly accessible and static benchmarks allow models to potentially tailor their responses to specific benchmark metrics, thereby inflating their performance. Addressing these limitations, we introduce a new benchmark **NPHardEval**. It contains a broad spectrum of 900 algorithmic questions belonging up to the NP-Hard complexity class, offering a rigorous

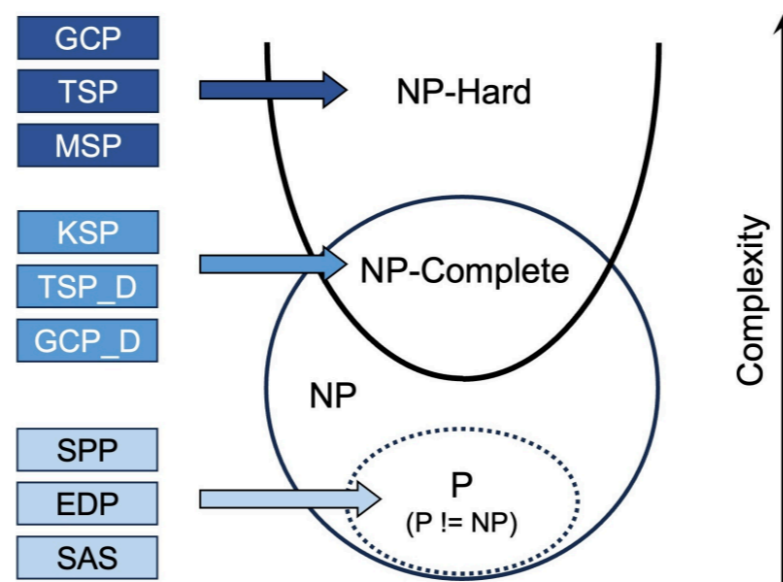
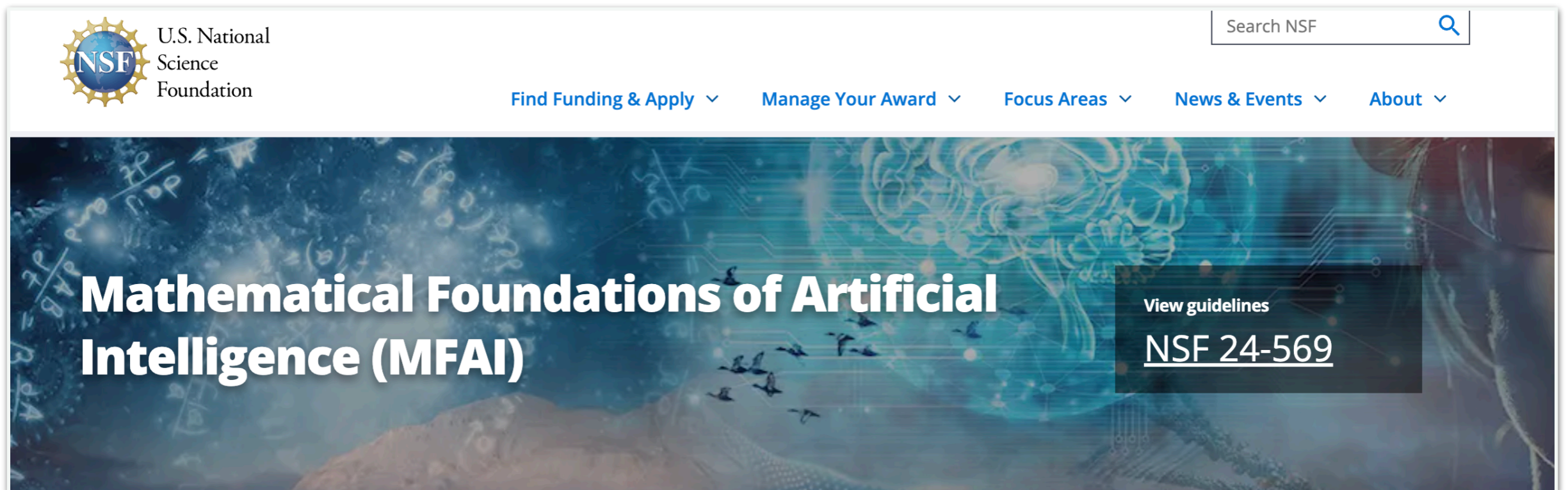


Figure 1: Computational complexity classes P, NP-complete, and NP-hard and corresponding tasks

"While fine-tuning yields improvements in solving polynomial time problems, its impact on the more complex NP-complete and NP-hard problems are negative. This suggests the inherent difficulty of hacking NP-complete, and potentially NP-hard, problems through the basic fine-tuning with question-and-answer approach."

# What Problems can LLMs Solve?

- We don't know yet but likely quite restricted (well within **P**)
- Growing need for mathematical foundations of AI



The image shows a screenshot of the NSF website. At the top left is the NSF logo with the text "U.S. National Science Foundation". To the right is a search bar labeled "Search NSF". Below the logo and search bar are navigation links: "Find Funding & Apply", "Manage Your Award", "Focus Areas", "News & Events", and "About". The main banner features a blue and teal background with mathematical symbols and a brain graphic. The text "Mathematical Foundations of Artificial Intelligence (MFAI)" is prominently displayed in white. A dark box on the right side of the banner contains the text "View guidelines" and "NSF 24-569".

U.S. National Science Foundation

Search NSF

[Find Funding & Apply](#) [Manage Your Award](#) [Focus Areas](#) [News & Events](#) [About](#)

**Mathematical Foundations of Artificial Intelligence (MFAI)**

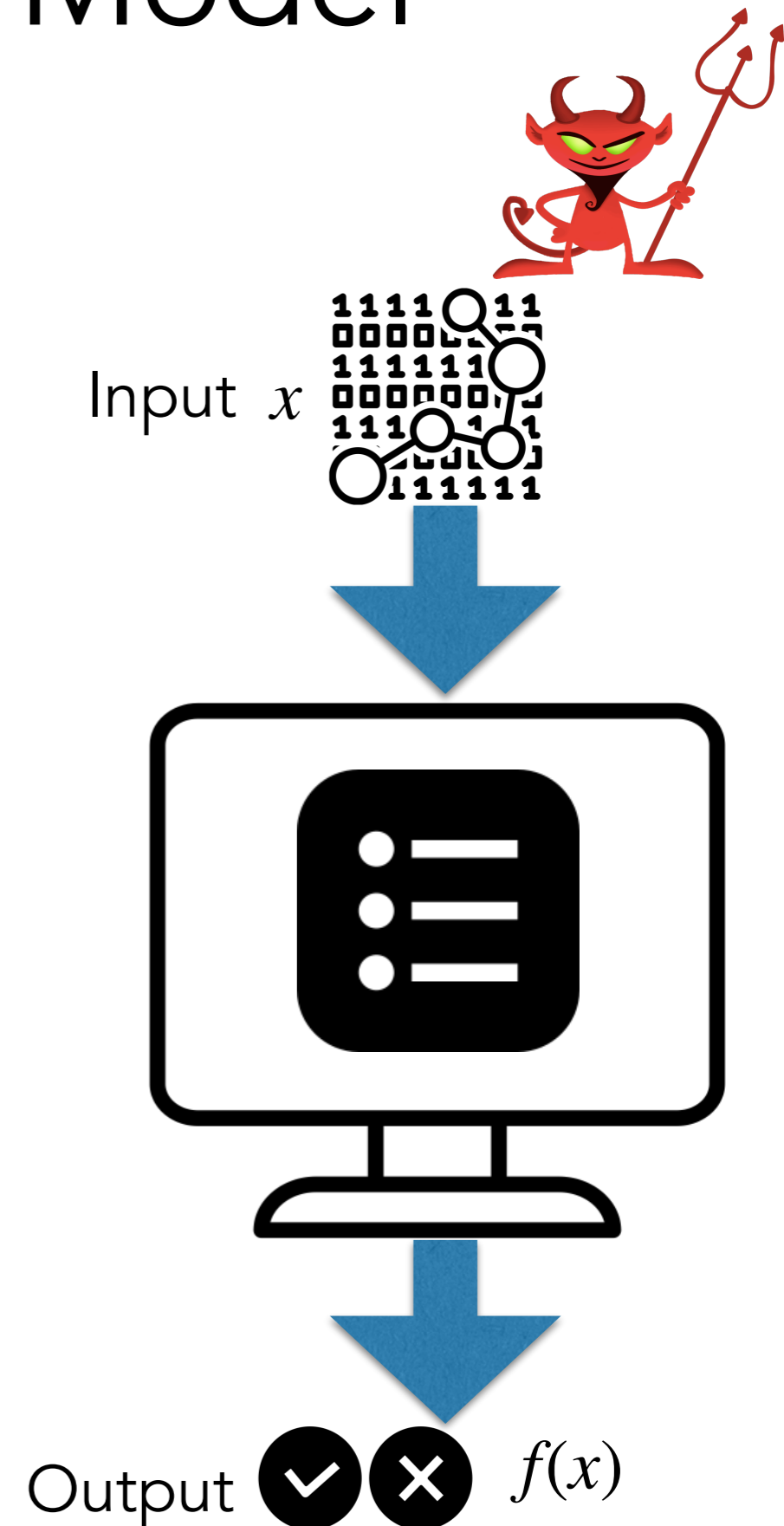
[View guidelines](#)  
[NSF 24-569](#)

**[Theory CS]**

Machine-Learning Augmented Algorithms

# Algorithm Analysis Model

- For any algorithm, some inputs are easy, others are hard
- **Worst-case paradigm:** Performance measured as the **maximum number of steps** taken on any input of size  $n$ 
  - Adversarial analysis



# Why Give Worst Case Guarantees?

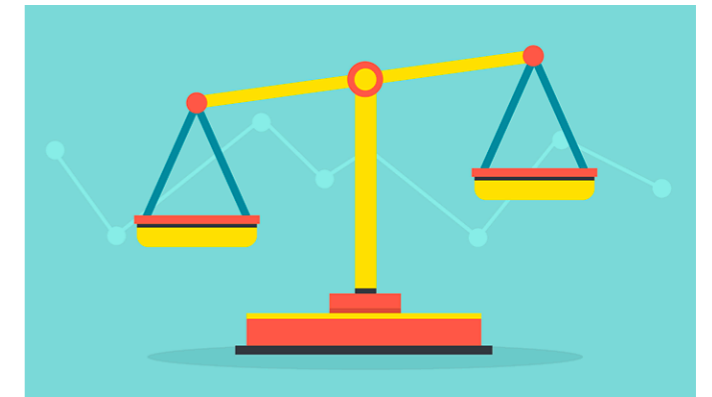
- Worst-case analysis: dominant algorithm design paradigm



Powerful guarantee



Universal



Mathematically  
compare algorithms

- **Downside:** Can often be too pessimistic, not predictive of performance on typical instances (in practice)



# Uninformed vs Informed Optimization

- Worst-case algorithms cannot take advantage of domain-specific structure: start from scratch every time
- ML heuristics do a lot better taking advantage of correlations in input
  - Downside: little or no guarantees



# Algorithms with Predictions

- Combines ML predictions with worst-case analysis
- Best of both worlds:
  - Do well when predictions are good
  - Be prepared for the worst-case when predictions are bad



# Space Complexity

# Space Complexity

- Space complexity  $f(n)$  of a deterministic Turing machine  $M$  is the maximum number of tape cells that  $M$  "scans" during its computation on any input of length  $n$
- Space complexity of a non-deterministic Turing machine is defined as the maximum number of tape cells that  $M$  "scans" during on any branch of its computation on any input of length  $n$

$\text{SPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space deterministic Turing machine}\}.$

$\text{NSPACE}(f(n)) = \{L \mid L \text{ is a language decided by an } O(f(n)) \text{ space nondeterministic Turing machine}\}.$

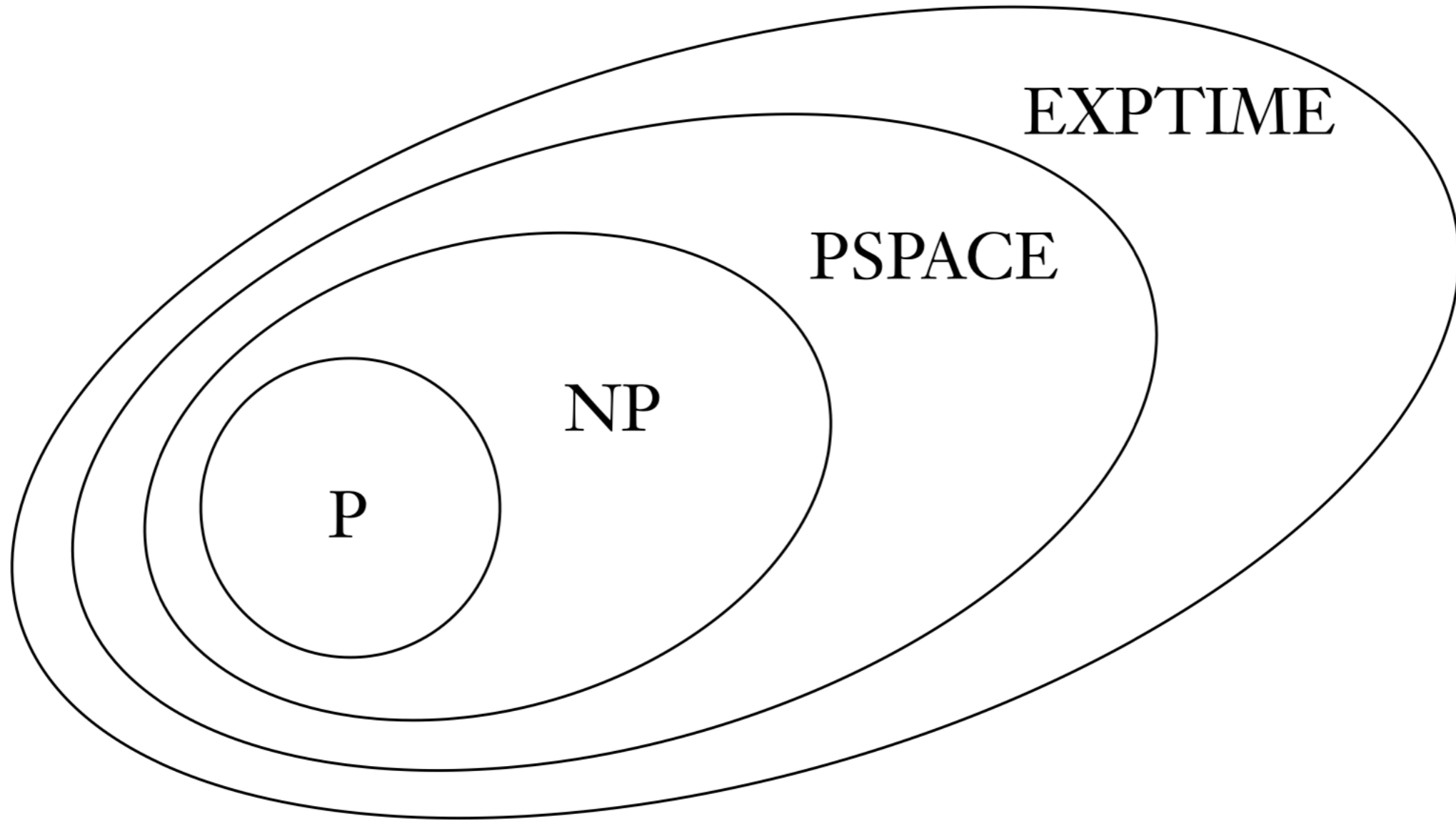
# Space Complexity

- $\text{PSPACE} = \bigcup_k \text{SPACE}(n^k)$
- $\text{NPSPACE} = \bigcup_k \text{NSPACE}(n^k)$
- **Theorem.**  $\text{PSPACE} = \text{NPSPACE}$

$\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXPTIME}$ .

We know  $\text{P} \neq \text{EXPTIME}$ , so one of these containments is proper but we don't know which one

# What We Believe



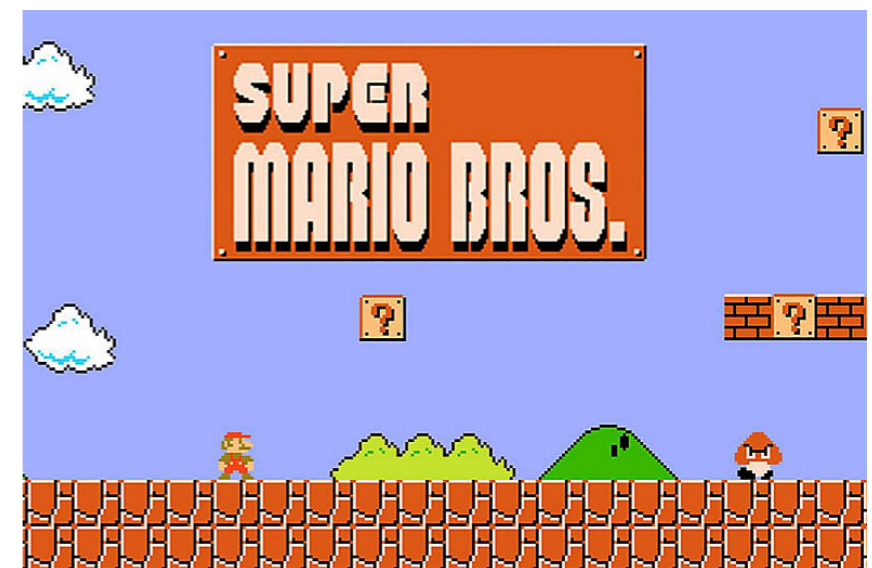
# PSPACE Complete Problem

- A language  $B$  is **PSPACE-complete** if it satisfies two conditions:
  - $B$  is in PSPACE, and
  - every language  $A$  in PSPACE is polynomial-time reducible to  $B$
- PSPACE complete problem:  
 $\text{TQBF} = \{\phi \mid \phi \text{ is a true quantified Boolean formula}\}$ 
  - Example of a quantified Boolean formula

$$\exists x_1 \forall x_2 \exists x_3 \left[ (x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\overline{x_2} \vee \overline{x_3}) \right].$$

# PSPACE and 2-Player Games

- PSPACE is essentially the complexity class of two-player games of perfect information
- Most games are PSPACE complete





# Generalized Geography: PSPACE Complete

- Two player game: players take turns to name cities
- Rules:
  - each city chosen must begin with the same letter that the previous city ended with
  - city names cannot be repeated
- Whenever a player cannot name a city (gets stuck), the other player wins
- Problem: Given a graph representation of possible moves and a starting node, does player 1 have a winning strategy?
- Like a two-player Hamiltonian cycle game