22.2
NP: Nondeterministic polynomial time

22.2.1
Introduction

P and NP and Turing Machines

1. **P**: set of decision problems that have polynomial time algorithms.
2. **NP**: set of decision problems that have polynomial time non-deterministic algorithms.

- Many natural problems we would like to solve are in $\boldsymbol{NP}$.
- Every problem in $\boldsymbol{NP}$ has an exponential time algorithm
- $\boldsymbol{P} \subseteq \boldsymbol{NP}$
- Some problems in $\boldsymbol{NP}$ are in $\boldsymbol{P}$ (example, shortest path problem)

Big Question: Does every problem in $\boldsymbol{NP}$ have an efficient algorithm? Same as asking whether $\boldsymbol{P} = \boldsymbol{NP}$.

Problems with no known polynomial time algorithms

Problems

1. **Independent Set**
2. **Vertex Cover**
3. **Set Cover**
4. **SAT**
5. **3SAT**

There are of course undecidable problems (no algorithm at all!) but many problems that we want to solve are of similar flavor to the above.

Question: What is common to above problems?

Efficient Checkability

Above problems share the following feature:

Checkability

For any YES instance I_X of X there is a proof/certificate/solution that is of length poly($|I_X|$) such that given a proof one can efficiently check that I_X is indeed a YES instance.

Examples:

1. **SAT** formula φ: proof is a satisfying assignment.
2. **Independent Set** in graph G and k: a subset S of vertices.
3. **Homework**

Efficient Checkability

Above problems share the following feature:

Checkability

For any YES instance I_X of X there is a proof/certificate/solution that is of length poly($|I_X|$) such that given a proof one can efficiently check that I_X is indeed a YES instance.

Examples:

1. **SAT** formula φ: proof is a satisfying assignment.
2. **Independent Set** in graph G and k: a subset S of vertices.
3. **Homework**

Sudoku

			2	5				
	3	6		4		8		
	4					1	6	
2								
7	6						1	9
								3
	1	5					7	
		9		8		2	4	
				3	7			

Given $n \times n$ sudoku puzzle, does it have a solution?

Solution to the Sudoku example...

1	8	7	**2**	**5**	6	9	3	4
9	**3**	**6**	7	**4**	1	**8**	5	2
5	**4**	2	8	9	3	**1**	**6**	7
2	9	1	3	7	4	6	8	5
7	**6**	3	5	2	8	4	**1**	**9**
8	5	4	6	1	9	7	2	**3**
4	**1**	**5**	9	6	2	3	**7**	8
3	7	**9**	1	**8**	5	**2**	**4**	6
6	2	8	4	**3**	**7**	5	9	1

THE END

...

(for now)