

# COS 126 Precept

April 21, 2009

# Agenda

---

- ▶ Exam 2 Information
- ▶ Universality/Computability
- ▶ Intractability
- ▶ Jeopardy
- ▶ Combinatorial Circuits



# Exam 2 Information

- ▶ **When:**

- ▶ Tuesday April 28<sup>th</sup> @ 10/11am

- ▶ **Where:**

- ▶ Written

- ▶ 10am class: McCosh 50
- ▶ 11am class: Friend 101

- ▶ Programming

- ▶ In Precept

- ▶ **Q & A Session**

- ▶ Monday April 26<sup>th</sup> at 7:30–9:30 in Friend 101
- ▶ For Alternate Exam Times:
- ▶ E-mail Donna Gabai ([dgabai@cs.princeton.edu](mailto:dgabai@cs.princeton.edu))

# Universality and Computability

## ▶ Universality

- ▶ All general purpose computers are equivalent to Turing Machines
- ▶ Turing Machine can do anything that can be described by a physically harnessable process of the universe

## ▶ Computability

- ▶ There are some problems that are inherently unsolvable
  - ▶ Halting Problem
  - ▶ General Virus Detection
  - ▶ Program Equivalence

# Intractability

- ▶ Deals with P, NP, NP-Completeness
- ▶ There exist some 'unsolvable' problems
  - ▶ Defy computation using Turing Machines
  - ▶ Our current model of computation
- ▶ But...
- ▶ We can categorize solvable problems\*
  - ▶ Based on computational requirements
  - ▶ Classification is independent of algorithm used to solve

\* classifying problems, not specific algorithms

# A Search Problem

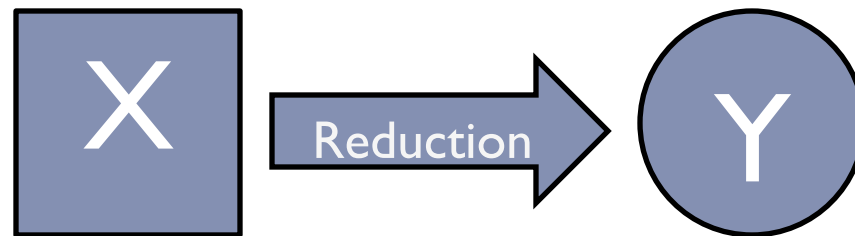
- ▶ Suppose we have a problem **I** that we want to solve
- ▶ Can we find a solution **S**, which can be verified (checked) in polynomial time?
- ▶ How long does it take to find this solution, **S**, relative to problem size?
  
- ▶ TSP is an example of a search problem
  - ▶ Trying to find shortest tour, must try all possible paths

# 3 Main Categories

- ▶ **P =**
  - ▶ {Search problems solvable in polynomial time on a deterministic Turing Machine}
- ▶ **NP =**
  - ▶ {Search problems verifiable in polynomial time}
  - ▶ Coming up with solution is HARD, but verifying it is easy
    - ▶ Factoring a number
- ▶ **NP-Complete = Hardest problems in NP**
  - ▶ If you can solve one problem in NP-Complete, then you can solve all NP problems
- ▶ **How is NP-Complete Possible?**
  - ▶ The idea of Reductions

# Reduction

- ▶ Idea: Relate difficulty of one problem to the difficulty of another.
- ▶ The **relative** difficulty of problem X to problem Y
- ▶ Says nothing about absolute difficulty of X

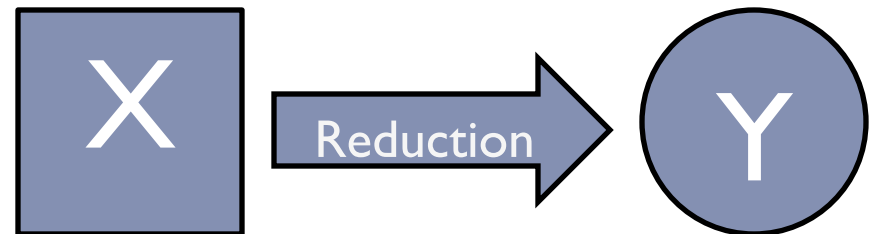


- ▶ Convert instance of problem X into instance of Problem Y
- ▶ This shows: X is no harder than Y
  - 8 ▶ It may be easier, but is no harder

# Details: Idea of Function Composition

- ▶ Reduction must take polynomial time
  - ▶ If it takes exponential, then reduction is in NP
- ▶ Suppose Solving problem Y takes  $g(n)$  time, Conversion of problem X into instance of problem Y takes  $f(n)$
- ▶ We can solve x in sum of times

$f(n) + g(n)$  is still in P  
 $f(g(n))$  is still in P



- ▶ Shows that X is no harder than Y
  - ▶ So, X is at most polynomial factor slower

# Examples of NP Problems

## ▶ Factoring

### ▶ Find Solution:

- ▶ Find factors of 23244

### ▶ Go!

### ▶ Verify Solution:

- ▶ Does  $23244 = 2^2 \times 3 \times 13 \times 149$
- ▶ Easy

## ▶ 3-SAT

### ▶ Find Solution to: $(x_1 \vee x_3 \vee \text{FALSE})(x_1' \vee x_3' \vee x_4)$

### ▶ Go!

### ▶ Verify Solution:

- ▶  $x_3 = \text{False}$
- ▶  $x_1 = \text{True}$

- 10 ▶ Easy

# Combinatorial Circuits

# Boolean Algebra

## ▶ 3 Operations: AND/OR/NOT

Operation	Meaning	Java Syntax	Common Notation
AND	X AND Y	X && Y	XY
OR	X OR Y	X    Y	X + Y
NOT	NOT X	!X	X'

# Boolean Algebra

## ► Laws and Identities

Law	Example
Commutative	$X + Y = Y + X$
Distributive	$(X+Y)Z = XZ + YZ$ $(X')' = X'' = X$
Associative	$(X+Y) + Z = Z + (Y + Z)$
DeMorgan's	$(AB)' = A' + B'$

# Truth Tables

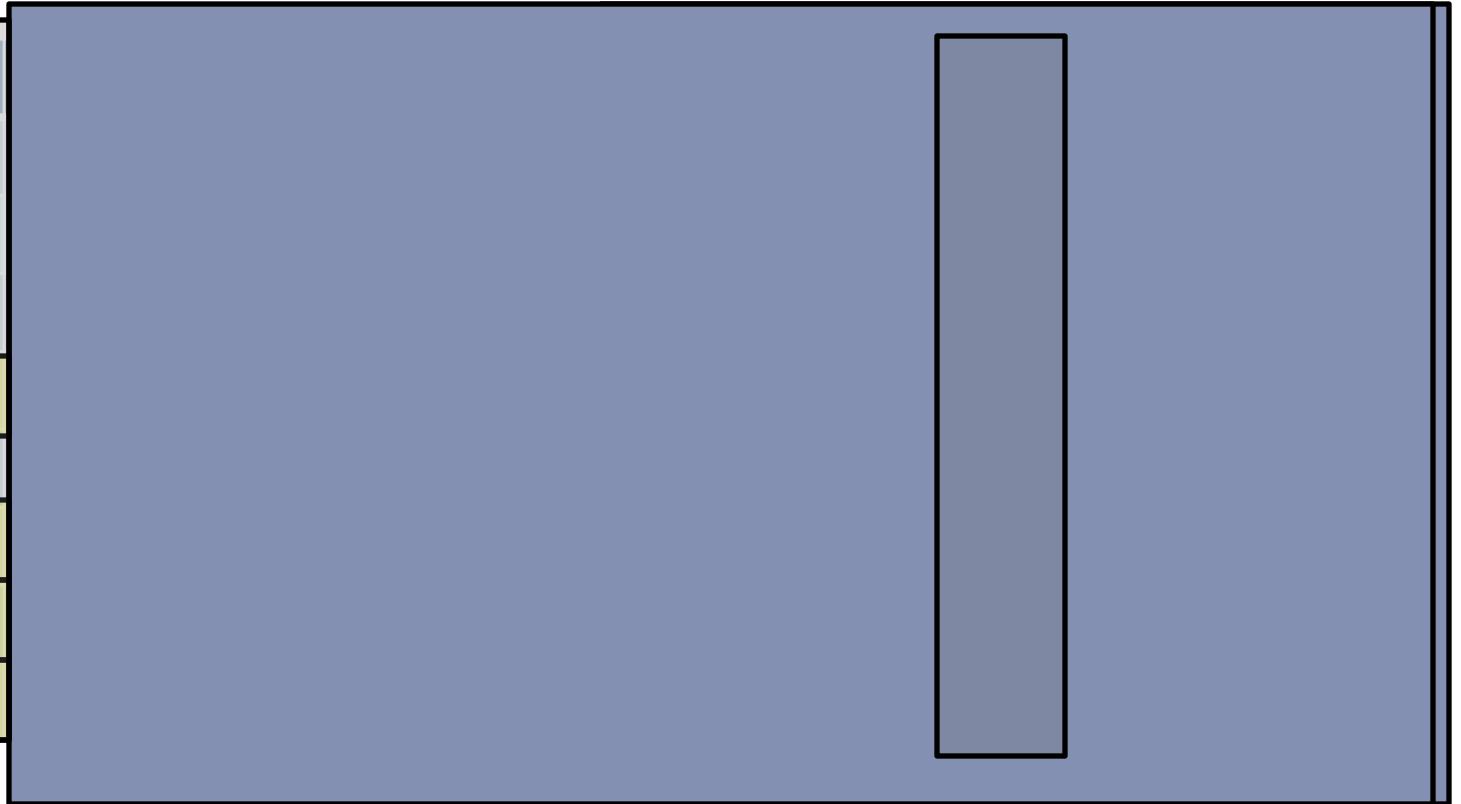
## ▶ Majority Function

- ▶ 3 inputs: X,Y,Z
- ▶ 1 Output: MAJ
  - ▶ Only a 1 if Majority of X,Y,Z are 1

$x$	$y$	$z$	MAJ
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

# Build Boolean Function

X	Y	Z	MAJ
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



Sum of Products

# Jeopardy – Universality

- ▶ Simple machine with tape for input/output and state transitions based on input
  - ▶ Turing Machine
- ▶ Thesis which stated: “...machines that can do anything that can be described by any physically harness able process of this universe.”
  - ▶ Church–Turing Thesis
- ▶ Designer of simple theoretical machine as ‘powerful’ as today’s computers
  - ▶ Alan Turing

# Jeopardy – Computability

- ▶ Problems for which no algorithm can be discovered
  - ▶ Unsolvable/Uncomputable/Undecidable
- ▶ Problem whose goal is to determine whether a Turing Machine will exit on a given input
  - ▶ Halting Problem
- ▶ To prove the halting problem one uses one of these logical constructions
  - ▶ Proof by contradiction/paradox

# Jeopardy – Intractability

- ▶ Set of all decision problems solvable in polynomial time
  - ▶ P
- ▶ The foremost outstanding question in theoretical CS
  - ▶  $P = NP$  ?
- ▶ The hardest types of problems checkable in polynomial time
  - ▶ NP-complete