# CS5800 – ALGORITHMS

# MODULE 2. COMPLEXITY (AND COMPUTABILITY & CRYPTOGRAPHY)

## Lesson 3: Computability

Ravi Sundaram

# Topics

- Basic philosophical issue
- Undecidability
  - Background
  - Concept
- Halting problem
- Landscape
- Summary

# Basic philosophical issue

- Is every well-posed mathematical problem decidable?
- Examples of well-posed mathematical problems
  - Does $x^4 + y^4 = z^4$ have solutions over the naturals?
  - Does there exist a polynomial-time factoring algorithm?
  - Does every digit occur in the same proportion in the decimal expansion of $\Pi$
- Examples of ill-posed or non-mathematical problems
  - Does god exist?
  - Can deep networks recognize cats?
  - Can water spontaneously explode?

# Undecidability

- For millennia philosophers have pondered about truth and provability
- They wished for a system of logic which would allow all true statements and only true statements to be proved.
- This dream was shattered by Kurt Gödel's tour de force argument in 1931 that showed no such system exists.
- In 1936 Alan Turing, building on work of Kurt Gödel and others, showed that the “halting problem is undecidable”.
- In addition to dramatically shortening Gödel’s argument this result laid the foundations of the modern computing revolution.

# Undecidability

- An undecidable problem is a decision(yes/no) problem for which it is proved to be impossible to construct an algorithm that always leads to a correct (yes-or-no) answer
- Alan Turing gave the first instance of an undecidable problem
- Halting problem – given a program P will P halt?
- Other examples:
  - Diophantine equation: does a given multivariate polynomial have an integral root?
  - Debugging: is a given program free of bugs?
  - Mortal matrix: given a collection of matrices is there a way to multiply them (possibly with repetition) in some order to get the zero matrix

# Halting Problem - Undecidable

- Halting problem - Given a program P does it halt?

Theorem: The halting problem is undecidable

Proof: By contradiction.

If decidable there exists a halting tester H that on any input P correctly says yes if P halts and no if it doesn't.

Consider the following program T:

Run H on T, i.e., execute H(T).

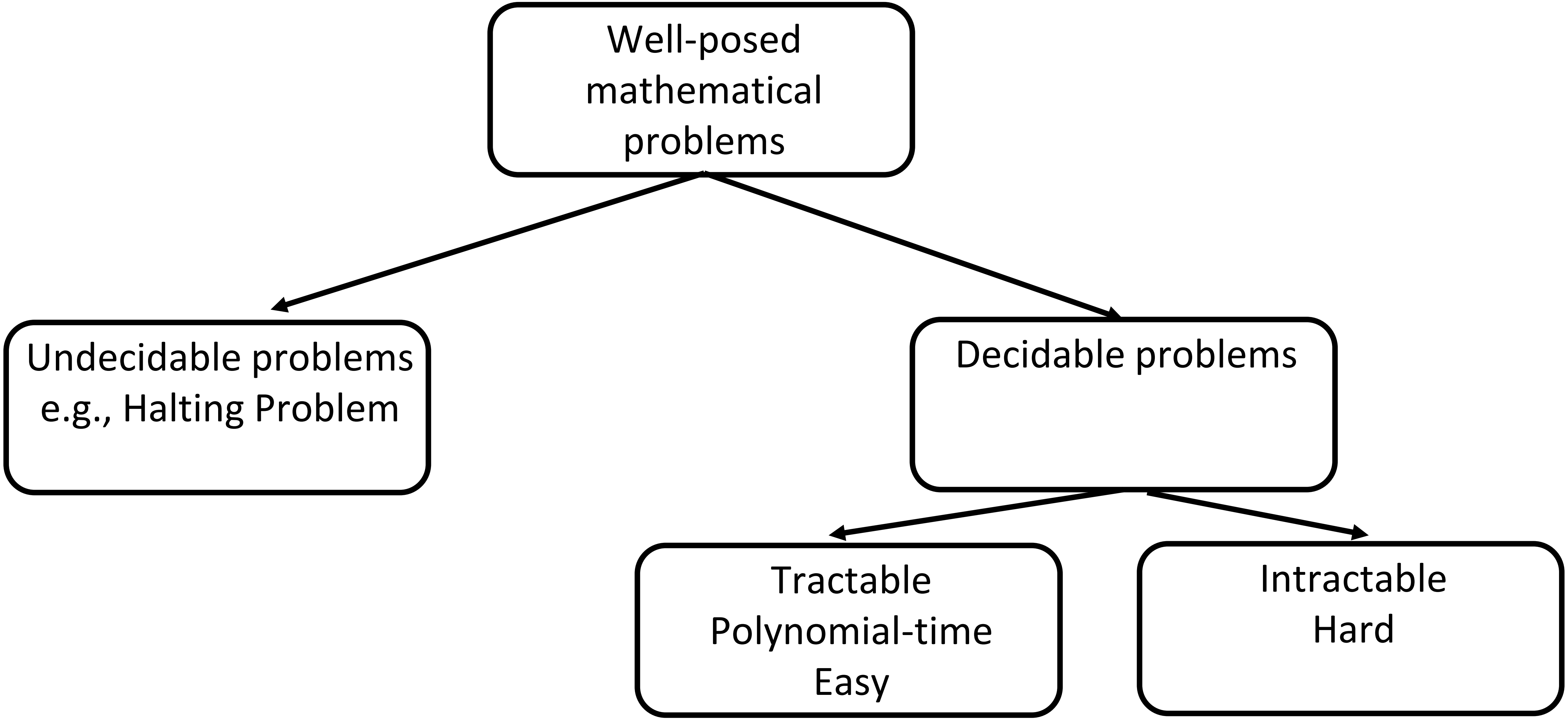
If H(T) says "yes" then go into an infinite loop

else halt

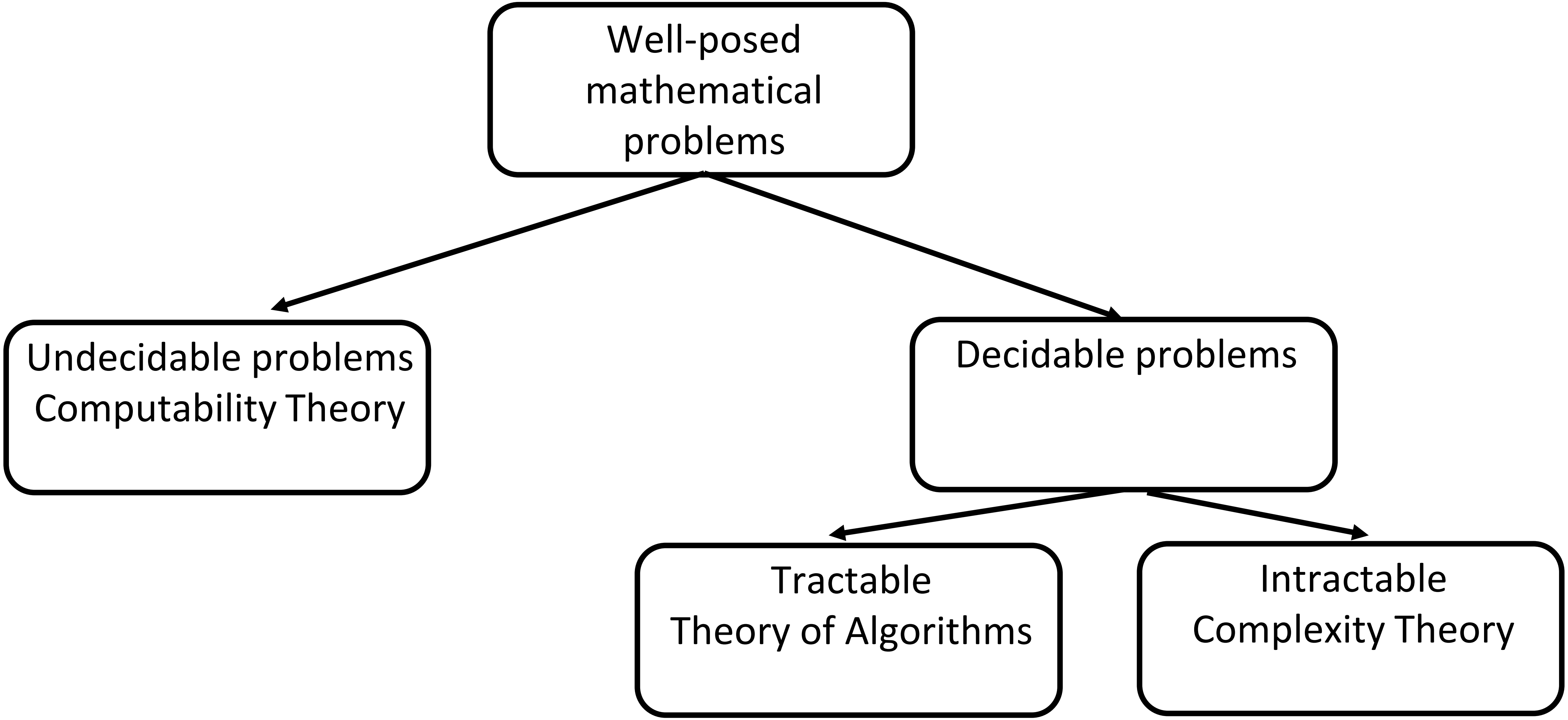
Observe that if H(T) says "yes" then T doesn't halt and if H(T) says "no" then T halts. Since we have derived a contradiction from the existence of H it must be the case that such an H cannot exist.

QED

# Landscape


Well-posed mathematical problems
Undecidable problems
e.g., Halting Problem
Decidable problems
Tractable
Polynomial-time
Easy
Intractable
Hard

# Landscape


Well-posed mathematical problems
Undecidable problems
Computability Theory
Decidable problems
Tractable
Theory of Algorithms
Intractable
Complexity Theory

# Summary

- Computability theory – study of undecidability
- Complexity theory – study of decidable problems and degrees of intractability
- Theory of Algorithms – study of tractable problems

- Cryptography
  - Crown jewel of Theoretical CS
  - Draws heavily from both  the easy and hard sides
  - Average case
  - Meta theorem – anything is possible