Today’s topics

- **Complexity**
  - *Great Ideas* Ch. 13
- **Computability**
  - *Great Ideas* Ch. 15, Brookshear Ch. 11

What can be computed

- **What class of problems can be solved?**
  - G4, 4 GHz Pentium IV, SGI Onyx, pencil?
  - Alan Turing proved some things, hypothesized others
    - Halting problem, Church-Markov-Turing thesis

- **What class of problems can be solved efficiently?**
  - Problems with no practical solution
    - what does practical mean?
  - Problems for which we can’t find a practical solution
    - solving one solves them all

Schedule students, minimal conflicts

- **Given student requests, available teachers**
  - write a program that schedules classes
  - Minimize conflicts

- **Add a GUI too**
  - Web interface
  - ...
  - ...

One better scenario

I can’t write this program because I’m too dumb

I can’t write this program because it’s provably impossible
Another possible scenario

I can’t write this program but neither can all these famous people

Not impossible, but impractical

- Towers of Hanoi
  - How long to move n disks?

- What combination of switches turns the light on?
  - Try all combinations, how many are there?
  - Is there a better way?

Travelling Salesperson

- Visit every city exactly once
- Minimize cost of travel or distance
- Is there a tour for under $2,000? less than 6,000 miles?
- Is close good enough?

Try all paths, from every starting point -- how long does this take?

a, b, c, d, e, f, g
b, a, c, d, e, f, g ...

Complexity Classifications

- This route hits all cities for less than $2,000 --- verify properties of route efficiently.
- Hard to find optimal solution

Pack trucks with barrels, use minimal # trucks

Ideas?

Problems are the “same hardness solve one efficiently, solve them all"
Are hard problems easy?

- **P** = easy problems, **NP** = “hard” problems
  - P means solvable in polynomial time
    - Difference between N, N^2, N^10?
  - NP means non-deterministic, polynomial time
    - guess a solution and verify it efficiently

- Question: P = NP? Rich or famous?
  - if yes, a whole class of difficult problems can be solved efficiently—one problem is reducible to another
  - if no, none of the hard problems can be solved efficiently
  - showing the first problem was in NP was an exercise in intellectual bootstrapping (1971)

Theory and Practice

- Number theory: pure mathematics
  - How many prime numbers are there?
  - How do we factor?
  - How do we determine primeness?

- Computer Science
  - Primality is “easy”
  - Factoring is “hard”
  - Encryption is possible

Important terms

- **Tractable**
- **Intractable**
  - Approximation?
- **Noncomputable**

Halt or not

- Does the following code eventually terminate?
  ```
  while (x > 1)
  {
    if (x > 2)
      x = x – 2;
    else
      x = x + 2;
  }
  ```
  - What if x is 8? How about 9?
Halt or not

- Does the following code eventually terminate?
  ```java
  while (x > 1) {
    if (x % 2 == 0)
      x = x / 2;
    else
      x = 3*x + 1;
  }
- What if x is 8? How about 7? How about all numbers > 0?
  ```

Consider this code

```java
// f is a filename of this program
if (DoesHalt(f,f))
  {
    while (true)
      { // do nothing forever
    }
  }
  return 0;
- We want to show writing DoesHalt is impossible
  - Proof by contradiction:
    - Assume possible, show impossible situation results
```

The halting problem: writing DoesHalt

```java
boolean DoesHalt(String proiname, String s)
  { // returns true if proiname halts given s as input, false otherwise
    if (DoesHalt(f,s)) cout << "does halt" << endl;
    else cout << "does not halt" << endl;
  }
- Programs that read program
- A compiler is a program that reads other programs as input
  - Can a word counting program count its own words?
- The DoesHalt function might simulate, analyze, ...
  - One program/function that works for any program/input
```

Noncomputable problems

- Halting Problem.
- Program Equivalence.
- Optimal Data Compression.
- Virus Identification.
- Impossible to write Java program to solve any of these problem!