Turing Machine, Limits of Computation
Class website

- http://www.soe.ucsc.edu/classes/cmps010/Spring11/

- Please write this down, and bookmark it

- Holds:
  - Syllabus (including homework due dates)
  - Homework assignment descriptions
  - Description of course readings
  - Links to class lecture notes

- The final exam is scheduled for Tuesday, June 7, 8am-11am
  - This class will have a final exam. Please plan on this.
Learning Support Services (LSS)

- Has tutoring available for students in CMPS 10
- Students meet in small groups, led by a tutor
- Students are eligible for up to one-hour of tutoring per week per course, and may sign-up for tutoring at https://eop.sa.ucsc.edu/OTSS/tutorsignup/ beginning April 5th at 10:00am.
- Brett Care - bcare@ucsc.edu is the tutor for CMPS 10 that LSS has hired
DRC Students

- If any student in the class requires a special accommodation for test taking or other assignment, please contact me
  - In person, or via email, ejw@cs.ucsc.edu
  - If you don’t contact me, I will not know you need this accommodation
  - The DRC office no longer sends notifications out about this
A little number review

- Integers
  - These are numbers that do not have a fractional part
  - They arise from counting things in the real world, or from subtracting one countable value from another

- Rational numbers
  - Numbers that are a ratio (a fraction) comprised of integers
  - Example: 1/9

- Irrational numbers
  - A number that cannot be expressed as the ratio of two integers
  - Examples include $\pi$ and $\sqrt{2}$
  - But, there are known techniques to compute these values
  - For example, in 2010 $\pi$ was computed to 4,152,410,118,610 places
Are some numbers uncomputable?

- In 1936, Alan Turing, a British mathematician, wondered if some numbers existed that were not computable?
  - That is, the number exists, since it can be placed on a continuous number line
  - But, no technique exists to compute this number to an arbitrary number of decimal places

- Why might someone be interested in this problem, especially in 1936?
  - It’s an unusual question to ask…
Principia Mathematica

- A three-volume work published in 1910-1912 by Alfred North Whitehead and Bertrand Russell
- Goal of this work was to **systematize mathematical knowledge**
- That is, to show that it is possible to:
  - Start from a small set of basic facts (axioms)
  - And then construct the entire edifice of mathematics …
  - Using a sequence of **mechanically applied**, logical rules.

*54·43. ⊢ a, β ∈ 1 . ∴ a ∨ β = ∆. ≡ a ∨ β ∈ 2

**Dem.**

\[ ⊢ *54·26 . \therefore a = t^x . β = t^y . ∴ a ∨ β ∈ 2 . ≡ x ∨ y. \]

[51·231] \[ ≡ . t^x ∨ t^y = ∆. \]

[51·1312] \[ ≡ . a ∨ b = ∆ \] (1)

\[ ⊢ (1) . *11·11·35 . ∴ \]

\[ ⊢ (a x, y) . a = t^x . β = t^y . ∴ a ∨ β ∈ 2 . ≡ . a ∨ β = ∆ \] (2)

\[ ⊢ (2) . *11·54 . *52·1 . ∴ \]

**Prop**

From this proposition it will follow, when arithmetical addition has been defined, that \( 1 + 1 = 2 \).

**Sequence of steps showing 1+1=2 from Principia Mathematica**
The goal of mechanical application of rules was central to Principia Mathematica
  - Implies that mathematics exists independent of human reasoning, is a purely abstract, logical construct

The existence of mathematical knowledge that could not be computed is therefore of great importance
  - Would mean that some part of mathematical knowledge **could not** be computed using mechanical rules

This is known as the decidability problem, and was part of the Entscheidungsproblem
  - “A quite definite generally applicable prescription is required which will allow one to decide in a finite number of steps the truth or falsity of a given purely logical assertion ...”
  - This was posed by David Hilbert in 1928

In 1935 Alan Turing was a masters student at King College, Cambridge, and learned about the Entscheidungsproblem...
Turing Machine

- Alan Turing determined that the question of whether some numbers are uncomputable provided a way of engaging the Entscheidungsproblem
  - He wrote about this in his paper, titled, “On Computable Numbers, with an Application to the Entscheidungsproblem”
    - [www.thocp.net/biographies/papers/turing_oncomputablenumbers_1936.pdf](http://www.thocp.net/biographies/papers/turing_oncomputablenumbers_1936.pdf)
- Of course, along the way he needed to develop a model of what purely mechanical computation looked like
  - In 1935, there were as yet, no electronic computers
  - To do this, he developed the **Turing Machine**
  - The Turing Machine is an abstract model of computation
  - It is a **thought** experiment
    - It didn’t matter that it wasn’t physically implemented
    - It just needed to be plausibly implementable using solely mechanical means (i.e., with no human intelligence involved in the operation of the machine)
Turing Machine: description

- A Turing Machine contains:
  - An infinitely long tape
    - Divided into cells
    - Each cell can hold a 1 or a 0, or be blank (B)
  - A head that can
    - Read or write a cell
    - Can move the tape left or right one cell (sometimes the head moves left or right)
  - A state register that holds the current value of the tape (the “state” of the machine, $q_i$)
  - A table, called the action table that holds a series of instructions
    - Based on the current state, $q_i$, and the value at the head, perform an action:
      - Either erase the cell, or write a 1 or 0 and also
      - Move the head left, write, or stay in position and also
      - Keep the same state, $q_i$, or update the state, (go to $q_{i+1}$)

- Video examples of Turing machines:
  - Mike Davey’s mechanical implementation: http://www.youtube.com/watch?v=E3keLeMwfHY
  - Aarhus University student implementation using Legos: http://www.youtube.com/watch?v=cYw2ewO6c4
Significance of Turing Machine

- **Anything** that is computable can be computed by a Turing Machine
  - With an appropriately constructed action table (program), a Turing Machine is capable of performing any feasible computation.
  - Hence, Turing Machine became an important analytical tool for determining **what is computable** and **what is uncomputable**.

- Turing demonstrated that there are many numbers that are not computable
  - In fact, it appears that there are perhaps more uncomputable numbers than computable ones
  - This seemed counterintuitive, since people had been focusing their attention on the computable numbers. It turns out these are the minority.
Significance of the Turing Machine (cont’d)

- With respect to the Entscheidungsproblem (Decidability problem), Turing found:
  - There is mathematical knowledge that cannot be computed
  - This means that there is mathematical knowledge that *cannot be found* via a mechanical application of logical steps
  - The answer to the Entscheidungsproblem is *no*
  - In combination with Godel’s incompleteness theorem, put the nail in the coffin of the goals of the Principia Mathematica project

- Oh, and as a side effect, created a strong theoretical model for thinking and reasoning about computation
Some problems are uncomputable

- The other major outcome of Turing’s work is a proof that there are some problems that a computer cannot solve
  - That is, even before the first electronic computer was created, Turing’s work established that there are limits to what can be accomplished with computation

- This seems very abstract. What is an example?
Post’s Correspondence Problem

- **Given:**
  - $N$ different card types
  - Each card type has two sequences of letters, one on top, one on bottom

- **Goal:**
  - Arrange the cards (can duplicate a card) so that the sequence of letters on the top, and on the bottom are the same (a yes instance), or
  - Report that it is impossible (a no instance)

- **Example:**
  - Given the following cards:
    - Here is a solution (cards: $1$, $3$, $0$, $2$, and a second copy of $1$)
Post’s Correspondence Problem

- Here is an example of a no instance:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>ABA</th>
<th>B</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>BAB</td>
<td>B</td>
<td>A</td>
<td>B</td>
</tr>
</tbody>
</table>

- Why is this a no?
  - In a yes, the top and bottom leftmost characters must be the same
  - This is impossible using the cards above
    - No cards have the same leftmost character on top and on bottom

- It is not possible to write a program that can solve this problem for an arbitrary set of cards
  - Why?
    - For some sets of cards, it will find a “yes” solution
    - But, if it cannot find a “yes” solution, since you can duplicate any card any number of times, the number of possible solutions is infinite.
      - Thus, it is not possible to exhaustively check all permutations
    - If each card could only be used once, then the problem would be computable, since we could look at every permutation (though this might take awhile)