CS 3313
Foundations of Computing:

Computation Models – a bit of history

http://gw-cs3313-2021.github.io

Who founded the field of Computer Science ?

1. Mark Zuckerberg
2. Brin and Page (google)
3. Bill Gates
4. Steve Jobs
5. von Neumann
6. Mauchly and Eckert (designers of Eniac)
7. ….???

Big questions/results in 20th Century & the people behind them

Proof as a procedure
Hilbert
1900

Provability?
Russell & Whitehead

Incompleteness Theorem
Godel
1931

Turing Machine
Alan Turing
1936

λ-Calculus
Church
1936

Computer Architecture
von Neumann
1944/45

NP-Completeness
Cook
1971

Concepts….

Solvable Problems

Computable Functions

Undecidable/Unsolvable

Encoding Machines as strings

Self replicating machines!

Computational Complexity/NPC

Hilbert's Problems

- *"Who of us would not be glad to lift the veil behind which the future lies hidden; to cast a glance at the next advances of our science and at the secrets of its development during future centuries?..... What new methods and new facts in the wide and rich field of mathematical thought will the new centuries disclose?.."* -- *Hilbert,1900* *at Int.Math Congress*
- Presented over 20 unsolved problems in mathematics
  - Including Cantor's Continuum hypothesis…
    - Concept of countable and uncountable infinite sets
- **2nd Problem: proof of consistency …what is a proof**
  - **Prove consistency of the axioms of arithmetic: effective procedure to prove any statement to be true or false.**
- 10th Problem: decidability: is there a universal algorithm for solving diaphantine equations?
  - Shown to be undecidable in 1970.
- Russell & Whitehead – "principia mathematica"
  - Logical bases for proving any statement…foundations of formal logic

Godel's Incompleteness Theorems

- Answer to Hilbert's 2nd Problem….
  - For any statement, can you generate a proof from a set of axioms?
  - *First incompleteness theorem…1931*
    - systems with properties of Peano arithmetic cannot be both complete and consistent
  - *Second incompleteness theorem*
    - no system with such properties can be proved to be consistent, unless it is inconsistent.
- Bottom Line: *There are statements (which can be expressed in predicate logic) that you can neither prove or disprove*
  - They may be true or false, but you cannot provide a proof for them.
- Want to learn more (easy to read): *Godel, Escher, Bach - An eternal Golden braid*, by Douglas Hofstadter
  - Was required reading in Theory of Comp Class!

Quest for Provability ends….Quest for Computability begins!

- Proving (true or false) every statement from a set of axioms was shown to be not possible…..
- Part of the question was "*find an effective procedure*"…so question now is *what is a procedure/algorithm* ?
  - What are the functions that can be "computed" using a finite set of discrete steps ?

- Quest was on for "model to define computation"
  - How to define "algorithm"
  - How to define function in a computable manner

Models of Computation…Turing Machine

- Alan Turing 1936: Turing Machine – a mathematical 'machine' model to recognize sets or compute functions
  - Finite set of states, semi-infinite tape for storage and input
  - How about functions on natural numbers ?
    - Encode the number in unary… n represented as 0^n (n zeros)
- Church 1936: λ-Calculus – a formalism to define and compute functions
  - Viewed as the "original" model for functional programming
    - LISP (Haskell) looks a lot like λ-calculus
- Other models: 1931 Recursive function theory – Godel
  - smallest class of functions which is closed under composition, recursion, minimization
- **Church-Turing (hypo-)thesis: any computable function can be computed by a Turing machine**

Turing Machines and Computers

- A Turing machine implements one function….equivalent to one computer program
- General purpose computer (RAM model – random access machine) can execute any program sent to it…..
  - Input to Computer is a program (and its input), and computer executes that program.

- Enter the Universal Turing machine (UTM): Input is a Turing machine and its input *w*, and the UTM simulates the TM on the input *w*.
- Languages and Machines…what is going on ?
  - Input to a Machine is another machine…..encoded as a string
  - Languages = machines !!!!

Computability/Solvability/Decidability

- Question : So are there functions that cannot be computed or problems that cannot be solved ?
- Concept of Unsolvable/Undecidable problems….
  - There are problems that cannot be solved by Turing machines
    - There are problems that cannot be solved by today's model of von Neumann Computers !!
- Does this mean there is no model of computation that is more powerful than a turing machine ?...open problem!

Quantum Computing…is this the answer ?

- QC works on the model of Q-bits (quantum bits)
  - We have instances where a problem can be solved more efficiently (time complexity) on a quantum computer
- Does this mean QC can do “more” than TM ?
  - It is more efficient (time) than a TM
  - But right now the model of QC has not solved any undecidable problems
- Is there an alternate to a TM ? Ever…?
- Interesting result (*filtered into simpler terms*): Computational models based on classical physics will be Turing-complete.
  - So….??????

Another interesting question:
Self Replicating Machines – von Neumann

- Scenario : You want to set up a mining colony on Mars, but Mars is not hospitable for humans (radiation, oxygen,…) so set up a colony on Mars inhabited with Robots

- Problem: The robots will need maintenance, and to provide more robots we need to ”assemble” robots on Mars…by a Robot !!

Self Replicating machines

- Question: can a robot assemble another robot, with all the functionality it has ?
  - Can it replicate itself ?
  - Implies: Does it have a “code” that encodes all of its functions ?
- another example: Can a 3-D printer print itself ?

von Neumann's Self Replicating Machines

- von Neumann provided a formal automaton (Cellular automaton) to prove that this could be done!
  - Three 'components':
    1. Description of machine (i.e, encode as string!)
    2. Universal constructor that reads description and constructs the machine
       - Simulation ?
    3. Copy machine: that makes copies of any description

  - Interesting observation: *machines get encoded as strings….and the assembly machine itself is encoded as a string!*

- Question: If we were talking about humans, what is the answer to "is there a code that describes all our functions"?

Self Replicating Machines – von Neumann…

- Cool fact: von Neumanm studied self replication machines, and provided the concept of cellular automaton for self replication, in 1948/49….BEFORE DNA discovery by Watson and Crick (& Rosalind Franklin) !

Efficiency of Computations: Time and Space Complexity

- Cook's Theorem (1971) set up the formal structure for defining efficiency of algorithms
  - Time and space complexity
- Deterministic turing machine time (DTIME) and Non-deterministic TM time
  - P = polynomial deterministic, NP= polynomial non-deterministic
- NP-completeness result: shows that any problem in NP (non-deterministic polynomial time) can be reduced (i.e., simulated?) in polynomial time to the SAT problem (which is also in NP)
  - Led to new class of problems ..NP-Complete Problems
  - If a problem is NPC, then implication is that finding a polynomial time algorithm is highly unlikely

Who founded the field of Computer Science ?

1. Mark Zuckerberg
2. Brin and Page (google)
3. Bill Gates
4. Steve Jobs
5. von Neumann
6. Mauchly and Eckert (designers of Eniac)
7. ….???

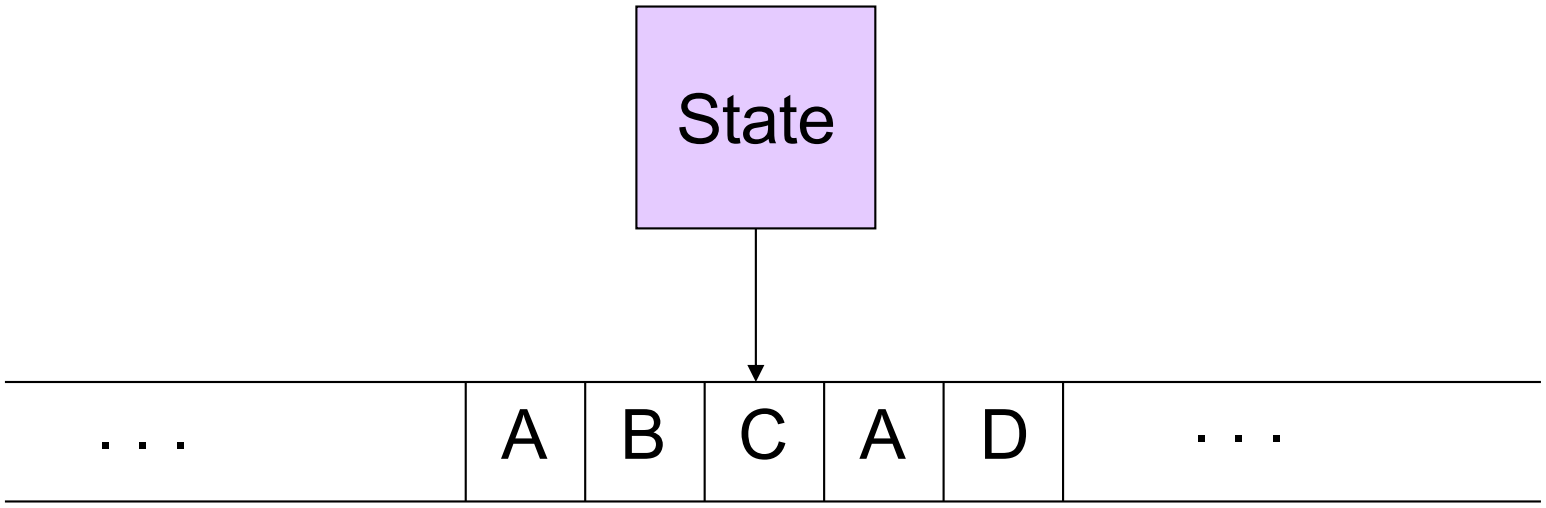
Moving on from PDAs….Turing Machines

- Finite State Automata (DFA/FSM): finite number of states
  - States store summary of past input/events
  - No external storage …so cannot have a counter to store variable
- Pushdown Automata (PDA): Add a “external” stack storage to a NFA
  - Single stack – first in-last out
  - What is stored in stack comes out in reverse when it is popped
- Extend PDA…..Two stacks, two-way input tape, etc…..OR
- Generalize the storage format to “random access”
  - Can store into any location and read from any location
  - Instead of a “box” as storage, we move to a (very long) shelf
- Turing Machine: NFA + external storage on a tape

Turing Machine

Action: based on the (i) state and (ii) the tape symbol under the read/write head:

- (1) change state, (2) write a symbol back to the tape and (3) move the head (left or right) one location/square on the tape.



Infinite tape with squares containing tape symbols chosen from a finite alphabet

Next..

- Turing Machine model
  - TM as an automaton
- Changing the basic TM model…..
  - Multiple tracks, multiple tapes, two-way tape/storage
  - Non-deterministic TM
- Equivalence of Deterministic and Non-deterministic TMs
  - Simulation procedure
- Simulation of RAM (Random Access Machine) on a TM
  - Simulation of von Neumann computer on a TM !!
- Universal Turing machine (encoding machines as strings)
- Solvable and Unsolvable problems…
- Time and space complexity
- Other models of computation: λ-calculus (functional programming)

Exam 2 Logistics & Info

- All material on Context free languages
  - (Homeworks 4-6, Quizzes 4-7)
- Exam will be posted on blackboard….your will write your answers and submit them (as one PDF file) using the Exam2-Submit link under assignments
  - Same as Exam1 logistics
- List of theorems and relevant results provided – under the Exam2-Submit link…you can refer to this source only
  - posted today.
- If you want partial credit (highly recommended ☺) ….
  - Provide a description of your Context free grammar or PDA in addition to the formal description (the CFG or transition function)
- Review all homeworks and the examples in the textbook