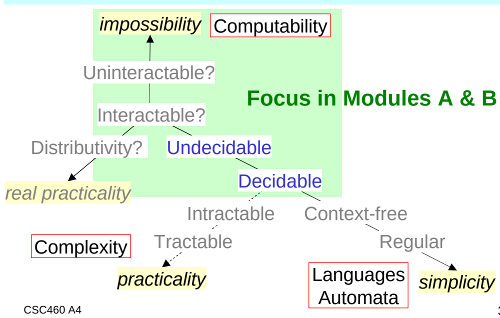
Questions/Review

- Ex A3
  - Part 1: Computational Problem Solving
    - Task 1: Program verification
    - Task 2: Map coloring
  - Part 2: Review
  - Part 3: Evaluation form and supporting notes
- Schematic representation
  - Set representation of a problem
  - Analyzing the set regarding Theory properties

Unit A4: Overview

- Overview: “Computability”
  - Intuition first; reasoning later (Module B)
- Introduce Turing machines (TMs)
  - Definition, properties, behaviors
- Learn how to operate TMs
- Preview Exercise A4 “Test-Drive TMs”
  - Using the JFLAP simulator

Overview: Theory of Computation

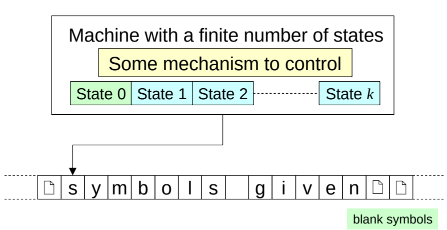


Tools for Computability Analysis

- Need for a standard abstract mechanism of computing sets
- Some possibilities
  - Some modern programming language
  - Some assembly language
  - Some kind of pseudo code
  - Mathematical functions
  - Other

Turing Machine (TM)

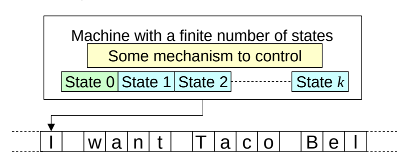
- Proposed by Alan Turing (1936)



Digression: more about Turing

Example Processes

- Word-by-word translation from English to Spanish, e.g., “I want Taco Bell”
- Detect palindromes
- Detect binary numbers



Additional Properties

- The tape has no left or right end.
  - Tape positions can be read/written.
  - The tape head can move left/right.
  - A tape position may be blank (□). The positions outside the input are filled with blanks.
  - The initial tape position is the leftmost input position.
- There are finite number of states.
  - There is a unique start state.
  - There are designated accept states.
- The input/tape alphabet contains a finite number of symbols.

TM Input

- Any string (of a designated alphabet)
- One instance of the complete information to be tested for its acceptability (cf. yes-no)
- Example
  - For the problem $\{(x, y, z) \mid x + y = z, \text{ where } x, y, z \in \mathit{NaturalNumbers}\}$, the following can be given as an input "...□1#2#3□...". A reasonable TM would accept this input. Cf. the binary representation: "...□1#10#11□..."

Possible TM Behaviors

- Halts
  - Accepts: If in one of the accept states
  - Dies: Otherwise
- Loops

Note: Assume no looping from an accepting state

- Information left on the tape
  - Often, ignored (side effects)
  - Possible to interpret it as the output

TM's Yes-No Response

- Option 1 (ternary)
  - Yes: If accepts (of course, halting)
  - No: If dies (still halting)
  - Don't know: If looping
- Option 2 (binary)
  - Yes: If accepts (of course, halting)
  - Not yes: Otherwise (i.e., no distinction between dying vs. looping)

Note: Possibly no response

TM Variations

- Single accept state
- Both accept/reject states
- Semi-infinite tape
  - I.e., the existence of the left end
- Multiple tapes
- Nondeterministic transition
  - I.e., δ as a relation, not a function

Cf. Sipser 1997: Semi-infinite tape, single accept, single reject

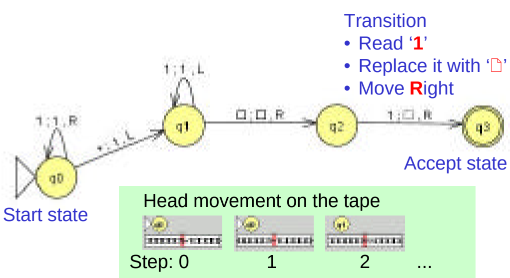
TMs vs. Computers

- Similarities and differences?

Interim Summary

- TMs are a simple, intuitive, and precise model of computation.
- The behavior of a TM is to capture the notion of "computation," also referred to as "effective procedure."
- When an effective procedure is guaranteed to terminate, it is called "algorithm."

Transition Diagram (JFLAP)



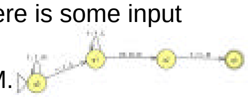
Formal Definition

TM $M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$

- Q: finite set of states
- Σ: finite set of input symbols, $\Box \notin \Sigma$ (blank symbol)
- Γ: finite set of tape symbols, $\Box \in \Gamma$, $\Sigma \subseteq \Gamma$
- δ: transition function $Q \times \Gamma \rightarrow Q \times \Gamma \times \{Left, Right\}$
  - E.g., $\{((q_1, 0), (q_2, 1, R)), ((q_2, 1), (q_3, \Box, L)), ...\}$
- q_0: the start state $q_0 \in Q$
- $\Box$: blank symbol (B in the text)
- F: set of accept states $F \subseteq Q$

Practice: Detecting Input

- Draw a transition diagram of a TM that would accept when there is some input (limit to 0 or 1).
- Formally define the TM.



TM $M = (Q, \Sigma, \Gamma, \delta, q_0, \Box, F)$

- Q: finite set of states
- Σ: finite set of input symbols, $\Box \notin \Sigma$
- Γ: finite set of tape symbols, $\Box \in \Gamma$, $\Sigma \subseteq \Gamma$
- δ: transition function $Q \times \Gamma \rightarrow Q \times \Gamma \times \{Left, Right\}$
- q_0: the start state $q_0 \in Q$
- $\Box$: blank symbol (B in the text)
- F: set of accept states $F \subseteq Q$

"Language" Examples

- Any sequence of 0's and 1's with exactly one '#': $\{x\#y \mid x, y \in \{\mathbf{0}, \mathbf{1}\}^*\}$
- Any sequence of 0's and 1's repeated after exactly one '#': $\{w\#w \mid w \in \{\mathbf{0}, \mathbf{1}\}^+\}$
- Any distinct sequences of 0's and 1's delimited by '#': $\{\#x_1\#x_2...\#x_l \mid \text{each } x_i \in \{\mathbf{0}, \mathbf{1}\}^+ \text{ and } x_i \neq x_j \text{ for each } i \neq j\}$
- Any sequence of the same number of 0's and 1's in that order: $\{0^n1^n \mid n > 0\}$

Unit Summary

- Understand the basics of Turing machines
  - Description, properties, behaviors
  - Transition diagram, formal definition, JFLAP
- Start Exercise A4 "Test-Drive TMs" in groups, in the lab
- Summary question
  - If TMs did not exist, what would you use as a model of computation? Explain.
  - Questions/Comments/Suggestions