Module B Evaluation

Review your portfolio

- My comments are sporadic and scattered on Review Ex, Comprehensive Ex, Supp. Notes, and Reflective Essay.
- You are encouraged to clarify and discuss my comments.
- You can keep the folder till the next class; then, return it to me.

Unit B7: Overview

- Review
  - Comprehensive/Review Exercises, etc.
- Explore the notion of "computation" via the Church-Turing thesis
  - Discuss equivalence of different mechanisms

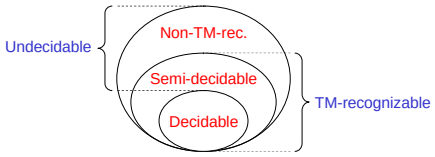
Correction on Eval Form B

4. The algorithmic notion of computation can be represented in a variety of equivalent forms, which define a bounded class of sets. That is, there is a limit to algorithmic computation. [**computability**]
   c. Can explain logically that the halting problem is semi-decidable and infinite-loop detection is ~~unsolvable~~ non-TM-recognizable.

Review

Computability Classes

- With respect to impossibility, problems can be classified into 3 disjoint sets, based on whether a TM exists and/or always gives an answer.



Module B Comprehensive/Review Exercise

- Required Problems: '$' Detection
  - **A**. To find out whether a given string contains at least one instance of the symbol '$'
  - **B**. To find out whether a given TM decides on Problem A
- Option 1: Virtual Memory Paging (Architecture)
- Option 2: Deadlock Prevention (OS)
- Option 3: Optimization (Compilers)
- Option 4: Protocol Security (Networks)
- Option 5: Semantic Errors (Databases)
- Option 6: Planning (AI)
- Option 7: Best Solution
- Option 8: Decidability Classes

Church-Turing Thesis

- Hypothesis as definition
  Computation/effective procedure [informal]
  = TM operation [formal]
- A variety of models of computation are equivalent to TM.
- All sorts of (algorithmic) computation are created equal [and do the same thing, computation].

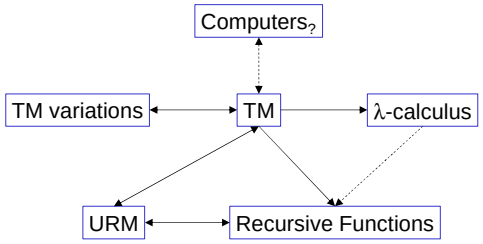
Take-home exercises

Connections

- TMs (including variants)
- Computers
  - Different programming paradigms
- Unlimited Register Machine (URM)
- Recursive functions
- λ-calculus [behind LISP]

Preview

Equivalence Summary



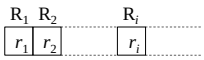
Proof of Equivalence

- Equivalence of models *M* and *N*
  - *N* simulates *M* *and* *M* simulates *N*
- Set identity, e.g., $A = B$
  - $A \subseteq B$ *and* $B \subseteq A$
- Equivalence of propositions, e.g., $p \leftrightarrow q$
  - $p \rightarrow q$ *and* $q \rightarrow p$
- Equality of values, e.g., $x = y$
  - $x \leq y$ *and* $y \leq x$

TM Variations

- Both accept and reject states (as well as other dying states)
- Semi-infinite tape
- Multiple tapes
- Nondeterministic transition
  - I.e., δ as a relation, not function

URM (1)



- Instructions:
  - Zero: $Z(i) \Rightarrow r_i \leftarrow 0$
  - Successor: $S(i) \Rightarrow r_i \leftarrow r_i + 1$
  - Transfer: $T(i, j) \Rightarrow R_j \leftarrow r_i$
  - Jump: $J(i, j, n) \Rightarrow$
    if $r_i = r_j$: proceed to the nth instruction
    otherwise: proceed to the next instruction

URM (2)

- Program: A finite sequence of instructions. Sequential execution of instructions except for possible jumps
- Initial values: Specific values can be assumed in the registers.
- Convention: The output in R_1.

Simulation by a TM?

Recursive Functions (1)

- Built up from the basic functions by a finite number of operations of substitution, recursion, and minimalization:
- Basic functions
  - Zero: 0
  - Successor: $x + 1$
  - Projection: $U^n_i (x_1, x_2, ..., x_n) = x_i$

Recursive Functions (2)

- Operations
  - Substitution (composition): Given computable functions $f(y1, ..., yk)$ and $g_1(x)$, ..., $g_k(x)$, define $h(x) = f(g_1(x), ..., g_k(x))$
  - Recursion: Given computable functions $f(x)$ and $g(x, y, z)$, define
    - $h(x, 0) = f(x)$
    - $h(x, y + 1) = g(x, y, h(x, y))$
  - Minimalization: Given computable function $f(x, y)$, define $h(x)$ = the least y such that $f(x, y) = 0$

 Simulation by a TM?

TM as Function

A TM can simulate a function. When accepts

- Function on strings
  - Input: A list of strings on the tape
  - Output: A string on the tape
- Function on natural numbers
  - Input: A list of natural numbers on the tape
  - Output: A natural number on the tape

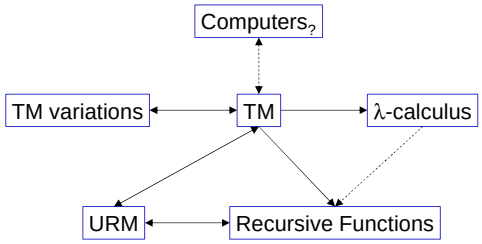
λ-calculus

Formal representation of functions

- Abstraction: $\lambda x.exp$
- Application: $exp_1\ exp_2$
- β-reduction: $(\lambda x.exp)\ y$
  - The result is exp where the instances of x is replaced with y.
  - E.g., $(\lambda x.f(f(x)))\ 2 = f(f(2))$ Is 2 a function?

In real life?

Equivalence Summary



TM vs. Computers

Equivalent?

Philosophy of Mind

- Hypothesis: Mind is a computer [Fodor: specifically uses TMs as the model].

Implications (if correct)?
Correctness?

Unit Summary

- Review
- Church-Turing thesis
  - Defines the informal notion of computation
  - Leads to the equivalence of various algorithmic computational approaches

Computability Summary

- Main goal: To be able to analyze the computability class and identify what we can do with algorithms [Comprehensive Ex]
- Topics
  - Problem as a set
  - Turing machines and (un)decidability
  - Mathematical tools: diagonalization, power set
  - Halting and other main problems; Rice's theorem
  - Reduction
  - Church-Turing thesis