Module B Evaluation

Review your portfolio

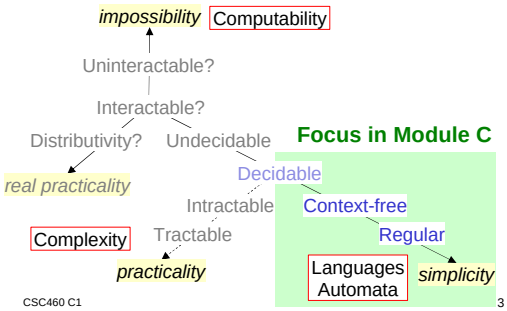
- My comments are sporadic and scattered on Review Ex, Comprehensive Ex, Supp. Notes, and Reflective Essay.
- You are encouraged to clarify and discuss my comments.
- You can keep the folder till the next class; then, return it to me.

Unit C1: Overview

- Overview Module C
- Discuss Ex B6/C0 "The Power of TMs"
- Introduce another model of computation (equivalent to the TM)
- Explore the effects of downgrading TMs
- Identifying the minimal mechanism for the given problem
- Preview Exercise C1 "Chomsky Hierarchy"

rough intro; more details later

Overview: Theory of Computation



Module C Goals (1)

Content Goals

5. Formal languages and automata
   a. "Hierarchy" of TM-recognizable languages
   b. Unrestricted grammars (rewriting system) ~ TMs
   c. Context-free languages (CFLs) ~ CFGs ~PDAs). When/how to use this class to analyze problems.
   d. Deterministic subset of CFLs ~ DPDAs
   e. Regular languages (regular sets~ RegExps~ FSAs). When/how to use this class to analyze problems.
   f. Show that some language is *not* regular.
   g. Show that some language is *not* context-free. That certain properties of CFLs are undecidable.
   h. Could explain this goal to CS students outside this class.

Module C Goals (2)

Content Goals

7. Power set
   a. Nondeterminism as the power set of the possible states.
   b. Nondeterminism with respect to TMs, PDAs, FSAs.

Performance Goals

4. Critical attitude
   a. *Critically* analyzed the usefulness of the "languages/automata" area of the (traditional) Theory of Computation
5. Communication

choosing the minimal mechanism

   a. Completed all the exercises on time (take-home and in-class).
   b. Made conscious efforts to promote transfer of learning among students

different background for this module

Ex B6/C0: The Power of TMs

The entire problem

- TMs are the appropriate mechanism.
- TMs are too weak.
- TMs are too powerful.

Modules/subproblems

Power-Economy Tradeoff

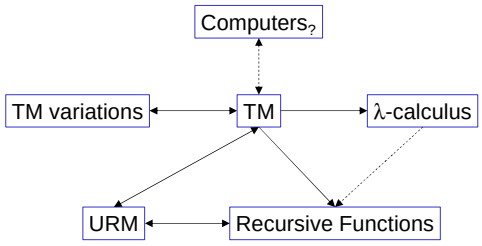
- Complex mechanisms are more powerful than simple ones (can represent more variety of problems/languages).
- Simple mechanisms are less expensive than complex ones (can operate simply and faster in general).
- An appropriate level would be the least powerful one for the requirement.

Downgrading TMs

- Imposing a specific tape organization
  - As a stack
  - As a queue
- Limiting the tape size
  - Set some finite fixed bound
  - Limit the growth (e.g., linearly with the input)
- Limiting the head mechanism
  - Read-only
  - One-way scanning only

Review

TM-Equivalent Models



Identifying Alternative Models

- Problem ~ set
  - Most commonly as a language (set of strings)
  - Recognized by a TM
- Whatever mechanism that can recognize (or generate) a string can be examined for the equivalence with a TM (or some downgraded forms).

Grammar

- Grammar [unrestricted grammar, semi-Thue system]
  - Rewrite rules of the form: $\alpha \rightarrow \beta$
    where α, β are strings of symbols and $|\alpha| > 0$

- $S \rightarrow NP_1\ V\ NP_2$
- $NP_1\ V\ NP_2 \rightarrow NP_2$ is Ved by NP_1
- $NP_1 \rightarrow$ liisa | tiina
- $NP_2 \rightarrow$ mikko | seppo
- $V \rightarrow$ kisses | kicks
- Ved $\rightarrow$ kissed | kicked

Using a Grammar

- Generating a string
- Accepting a string (parsing)

- $S \rightarrow NP_1\ V\ NP_2$
- $NP_1\ V\ NP_2 \rightarrow NP_2$ is Ved by NP_1
- $NP_1 \rightarrow$ liisa | tiina
- $NP_2 \rightarrow$ mikko | seppo
- $V \rightarrow$ kisses | kicks
- Ved $\rightarrow$ kissed | kicked

- Terminal symbols [lower case]: Cannot be rewritten
- Nonterminal symbols [UPPER CASE]: Must be rewritten

TM vs. Grammar

- Equivalent!
- I.e., both can recognize/generate exactly the same class of languages
  - TM-recognizable
  - Recursively enumerable (RE) [standard term in this context]
- Proof idea
  - A grammar simulated by a TM
  - A TM simulated by a grammar

TM or Grammar?

- Modeling computation
- Modeling function
- Classifying sets (including computational problems)
- Declarative specification
- Specification of programming languages
- Description of human languages
- Analysis of different classes

Grammar Variations

- Conditions on the length of LHS (left-hand side) and/or RHS
- Conditions on the type of string on the RHS

- $S \rightarrow NP_1\ V\ NP_2$
- $NP_1\ V\ NP_2 \rightarrow NP_2$ is Ved by NP_1
- $NP_1 \rightarrow$ liisa | tiina
- $NP_2 \rightarrow$ mikko | seppo
- $V \rightarrow$ kisses | kicks
- Ved $\rightarrow$ kissed | kicked

Grammar Hierarchy

- Type 0: Unrestricted Grammar
- Type 1: Context-Sensitive Grammar (CSG)
  - $|LHS| \leq |RHS|$ [i.e., no shrinking]
- Type 2: Context-Free Grammar (CFG)
  - $|LHS| = 1$ [i.e., ignore the context of rewriting]
- Type 3: Regular Grammar ~ Regular Expression
  - RHS has at most one nonterminal at the left/right edge.

examples?

Automata Hierarchy

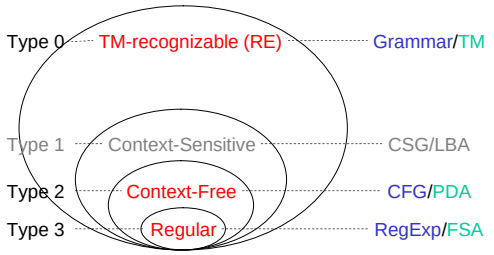
- Type 0: TM
- Type 1: Linear Bounded Automaton (LBA)
  - Tape space limited to the input size
- Type 2: Pushdown Automaton (PDA)
  - Tape as a stack (with a separate input tape)
- Type 3: Finite-State Automaton (FSA)
  - Head movement only to the right

schematics

Language Hierarchy

- Type n languages are accepted/generated by Type n grammars and recognized by Type n automata.
- Type 0: RE (TM-recognizable)
- Type 1: Context-Sensitive Languages (CSLs)
- Type 2: Context-Free Languages (CFLs)
- Type 3: Regular Languages

Chomsky Hierarchy



Contributions of Chomsky

Group Exercise 1

Identify the minimal type for the following

- Pattern matching?
- Programming language specification?
- Representing human language?

Discuss the following

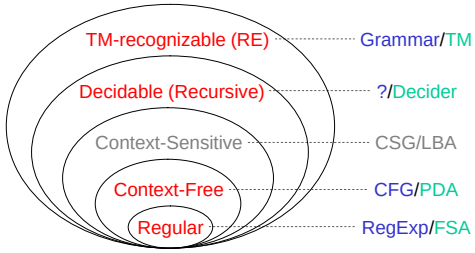
- Effects of nondeterminism?
- Is the hierarchy exhaustive?
  - Where is the set of decidable languages?
  - Class like Type 1.5?

See Ex C1

Recursive Sets

- Decidable = Recursive
- Theorem: CSLs $\subset$ Recursive Sets (i.e., proper inclusion)
- Proof idea
  - CSLs $\subseteq$ Recursive Sets
  - There is a recursive set that is not a CSL.

Extended Chomsky Hierarchy



Power-Economy Tradeoff, Revisited

- Higher-level grammars/automata are more powerful than lower-level ones (can represent more variety of languages).
- Lower-level grammars/automata are less expensive than higher-level ones (can operate simply and faster in general).
- An appropriate level would be the least powerful one for the requirement.

Unit Summary

- Grammars
- Connection between grammars, automata, and languages (problems)
- Chomsky hierarchy
- Identifying the minimal mechanism for the given problem

Summary Question

- What would be your guiding principle to identify the simplest mechanism/approach (not necessarily in CS)? Explain.
- Questions/Comments/Suggestions