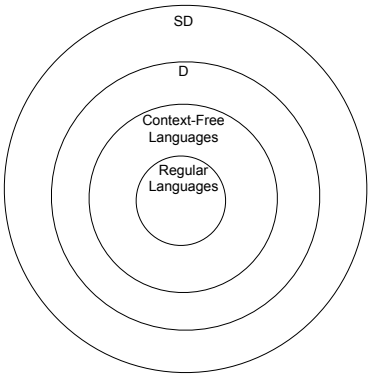
Lecture 32: Decidability & Semi-Decidability

CSCI 81
Spring, 2015

Kim Bruce

No Class Friday!
... but see recorded lecture!

Decidable & Semidecidable



The Hierarchy

- Theorem: The set of context-free languages is a proper subset of D.
  - Proof: Every context-free language is decidable, so the context-free languages are a subset of D.
  - There is at least one language, $A^nB^nC^n$, that is decidable but not context-free.

Distinguishing D and SD

- Most obvious languages in SD also in D
  - $A^nB^nC^n = \{a^nb^nc^n \mid n \geq 0\}$
  - $\{wcw \mid w \in \{a, b\}^*\}$
  - $\{ww \mid w \in \{a, b\}^*\}$
  - {w of form x*y=z: x,y,z ∈ {0, 1}* and, when x, y, and z are viewed as binary numbers, x*y = z}
- But already found some in gap, e.g. H_{TM}

Outside of SD

- Uncountably many languages outside of SD
- *Example:* Complement of H_{TM}

Closure Properties

- Theorem: D is closed under complement
  - Proof: Let L ε D. Build TM deciding L
  - ...
  - Proof depends on TM deterministic and always halts.
- What about SD?
  - Not true for H_{TM}

Equivalences to SD

- A TM M *enumerates* the language L iff, for some fixed state p of M,
  $L = \{w : (s, \varepsilon) \vdash_M^* (p, w)\}$.
  - *Potentially infinite computation.*
- A language is *Turing-enumerable* iff there is a Turing machine that enumerates it.

SD & Turing Enumerable

- Theorem: A language is SD iff it is Turing enumerable.
  - Proof: Spose L is Turing enumerable. Show L is SD.
    - Let w be input. Start enumerating L. Every time enter state p, check to see if contents of tape is w. If yes then halt and stop. Otherwise keep going.
  - "Obvious" proof in other direction not work!

Lexicographic Order

- Two definitions:
  - Dictionary order
  - Order by length, and then within elements of same length, order in dictionary order.
- We will always mean second, because can enumerate all elements of Σ^* in that ordering!

SD & Turing Enumerable

- Theorem: A language is in SD iff it is Turing enumerable.
  - Proof (*cont*): Spose L is in SD. Show L can be enumerated.
    - Enumerate all $w \in \Sigma^*$ lexicographically: ε, a, b, aa, ab, ba, bb, ...
    - As each w_i is enumerated, start a copy of M to check with w_i as input.
    - Execute one step of each M with w_i started, excluding those that have halted
    - Whenever an M accepts a w_i, output w_i.
- Called dove-tailing

Lexicographically Enumerable

- M *lexicographically enumerates* L iff M enumerates the elements of L in lexicographic order.
- A language L is *lexicographically Turing-enumerable* iff there is a Turing machine that lexicographically enumerates it.

Lexicographically Enumerable

- Theorem: A language is in D iff it is lexicographically enumerable.
  - Proof: Suppose L is in D. Show L can be enumerated lexicographically.
    - Enumerate all $w \in \Sigma^*$ lexicographically: ε, a, b, aa, ab, ba, bb, ...
    - As each w_i is enumerated, run M deciding L on w_i as input.
    - If M accepts a w_i, output w_i. Otherwise go to next.
    - Easier here because M always halts

Lexicographically Enumerable

- Theorem: A language is in D iff it is lexicographically enumerable.
  - Proof (*cont)*: Spose L can be enumerated lexicographically. Show L is in D.
  - To determine if w in L:
    - Start enumerating all elts of L lexicographically
    - If w is enumerated, then accept.
    - If go past w in lexicographic order, then reject
    - If halts before getting to w, then also reject.

Oops!

- Second part of proof has a hole.
  - Suppose M lexicographically enumerates L = {a,ba,aba} by enumerating three elements and then continuing forever without ever enumerating another element. If w comes after last element, then won't know if in or not!
  - Can only happen if L is finite.
  - But all finite languages are decidable.
  - Fixes proof
- But not decidable whether L is finite!!

Summary

- Church-Turing Thesis: all models of computation give same computable functions
- Halting problem is undecidable.
  - Can prove lots of others undecidable by using them to solve halting problem.
- Semi-decidable sets equivalent to Turing enumerable.
  - Decidable if Turing lexicographically-enumerable.

Next (recorded) lecture

- Using reductions to show problems are not decidable (or not semi-decidable).
- Rice's theorem: Every non-trivial problem about semi-decidable (i.e., TM-accepted) languages is undecidable!