CS 205
Lecture 32
5/4/21

Closure Properties of Recursive & RE languages

Recall
- L recursively enumerable (semi-decidable) : $L = L(M)$ for some TM M
- L recursive (decidable) : $L = L(M)$ for some TM M that halts on all inputs

Thm : Recursive languages are closed under complementation.

Proof : Suppose L is recursive, i.e. $L = L(M)$ and M halts on all inputs.

Machine $\bar{M}$ that accepts $\bar{L}$:

On input w, $\bar{M}$ runs M on w
$\bar{M}$ accepts w if M rejects, and
$\bar{M}$ rejects w if M accepts.

Thm : Recursive languages are closed under intersection.

Proof : Let M_1 and M_2 be TMs that accept L_1 and L_2 resp., and halt on all inputs.

TM M that accepts $L_1 \cap L_2$

On input w
  Run M_1 on w
  if M_1 accepts w then
    Run M_2 on w
    if M_2 accepts then accept $\boxed{\text{else reject}}$
  $\boxed{\text{else reject } w}$

Thm : RE languages are closed under intersection.

Union : $L_1 \cup L_2 = \overline{\bar{L}_1 \cap \bar{L}_2}$

Hence recursive languages are closed under union.

Thm : RE languages are closed under union.

Proof : Let $L_1 = L(M_1)$ and $L_2 = L(M_2)$.

TM M that accepts $L_1 \cup L_2$

On input w
  Dovetail the computations of M_1 and M_2 on w and accept if either M_1 or M_2 accepts

Alternatively

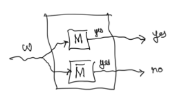
TM M : On input w
  Nondeterministically choose to run either M_1 or M_2 on w and accept if it accepts.

Thm : L is recursive iff L and $\bar{L}$ are both r.e.

Proof : ($\Rightarrow$) If L is recursive then L is r.e. Moreover, if L is recursive then so is $\bar{L}$, and hence $\bar{L}$ is r.e.

($\Leftarrow$) Suppose $L = L(M)$
$\bar{L} = L(\bar{M})$

TM M : On input w
  Dovetail the computations of M and $\bar{M}$ on w



Consequences
(1) L is not r.e. $\Rightarrow$ L is not recursive.
(2) $\bar{L}$ is not r.e. $\Rightarrow$ L is not recursive.
(3) L is r.e. but not recursive $\Rightarrow$ $\bar{L}$ is not r.e.

Decision Problems

Ex
- given a CFG G is G ambiguous?
- given a TM M, is $L(M) = \Sigma^*$?
- given DFAs M_1, M_2 is $L(M_1) = L(M_2)$?

A decision problem can be represented as a language membership problem.
- $\{\langle G\rangle \mid G \text{ is ambiguous}\}$ — language
  ↳ $\langle G\rangle$: a string encoding the CFG $G = (V, T, S, P)$
- $\{\langle M\rangle \mid L(M) = \Sigma^*\}$ — language
- $\{\langle M_1, M_2\rangle \mid L(M_1) = L(M_2)\}$

Undecidability

A decision problem is undecidable if there is no algorithm that solves the problem.

Algorithm : TM M that halts on all inputs and which accepts the language corresponding to the decision problem.

This means : A decision problem is decidable iff the corresponding language is decidable/recursive.

Encoding TMs as strings over $\{0,1\}$ $\Sigma = \{0,1\}$ input alphabet

TM M has :
- States : $q_1, q_2, \cdots, q_k$ (q_1 start state, q_2 unique accepting state)
- Tape symbols : $X_1, X_2, \ldots, X_m$ ($X_1 \to 0$, $X_2 \to 1$, $X_3 = B$)
- Directions L and R ($L \to 1$, $R \to 2$)

Code for the TM M :

$111 \underbrace{\quad\quad}_{\text{transition } 1} 11 \underbrace{\quad\quad}_{\text{transition } 2} 11 \cdots 11 \underbrace{\quad\quad}_{\text{last transition}} 111$

$$\delta(q_i, X_j) = (q_k, X_l, D_m)$$

encoded as

$$\boxed{0^i 1 0^j 1 0^k 1 0^l 1 0^m}$$