CS452 HW7

Yuting Chen

4.20

Let A and B be two co-turing recognizable languages.

⇒ There exists turing machines that recognizes the complement of these languages.

Let these be TM $M_{\bar{A}}$, TM $M_{\bar{B}}$.

$L(M_{\bar{A}}) = \bar{A}$, $L(M_{\bar{B}}) = \bar{B}$.

Assume $A \cap B = \phi \Leftrightarrow \bar{A} \cup \bar{B} = \Sigma^*$.

Construct a Machine M is a decider that separates A and B.

Bulid a Turing Machine that will do as follow:

M = "on input w.

1. Run machines $M_{\bar{A}}$ and $M_{\bar{B}}$ in parallel.
2. If $M_{\bar{B}}$ accepts w, output accept.
3. If $M_{\bar{A}}$ accepts w, output reject.

According to preliminaries we know that any string in Σ^* is either on $\bar{A}$ or $\bar{B}$.

M uses recognizers for $\bar{A}$ and $\bar{B}$

⇒ It will halt on every input

A and B are disjoints

⇒ Any string in A is in $\bar{B}$. $(A \leq \bar{B})$

Every string in $\bar{B}$ is accepted by M. ⇒ $A \leq L(M)$

If any string in B is in $\bar{A}$. $(B \leq \bar{A})$

But every string in $\bar{A}$ is rejected by M.

⇒ B is not in the language of M.

Thus, $L(M) = C$ is decider that separates A and B.

4.26

Assume $PAL_{DFA} = \{\langle M\rangle \mid M$ is a DFA that accepts some paliodrome.$\}$

If there is a Turing Machine can be presented for the given DFA that runs finitely and halts, then the PAL_{DFA} is decidable.

Construct a decider X for PAL_{DFA} and a Turing Machine Y that can decides E_{CFG}:

Bulid a Turing Machine that will do as follow:

X = " On input $\langle M \rangle$.

1. Construct a PDA P_1 such that $L(P_1) = \{w \mid w \text{ is a palindrome}\}$
2. Construct a PDA P_2 such that $L(P_2) = L(P_1) \cap L(M)$.
3. Convert P_2 into an equivalent CFG G.
4. Use Turing Machine Y to check if L(G) is empty.
5. If L(G) is empty then output reject.
6. Else, output accept.

For Turing Machine Y:

Step 1 and 2 can be done in finite steps. Step 3 takes finite steps to convert P_2 into its equivalent CFG. For Step 4, the decider Y checks if the language L(G) is empty or L(G) is not empty. It can be done in a finite step.

So, we got that X takes finite steps for any input, it's a decider.

Thus, PAL_{DFA} is decidable.

5.21

The string $t_{i_1} t_{i_2} \cdots t_{i_l} a_{i_1} a_{i_2} \cdots a_{i_l}$ has at least two derivations from T and B. When P has a match with $t_{i_1} t_{i_2} \cdots t_{i_l} = b_{i_1} b_{i_2} \cdots b_{i_l}$

If the CFG G is ambiguous, then some string s has multiple derivations.

Check the grammar G, it can be seen that derivation of T and B can each generate strings at most one string similar, let it be s.

The derivations of s will be as follow:

$S \Rightarrow T$. $s = t_{j_m} t_{j_{m-1}} \cdots t_{j_1} a_{j_1} a_{j_2} \cdots a_{j_m}$

$S \Rightarrow B$. $s = b_{j_m} b_{j_{m-1}} \cdots b_{j_1} a_{j_1} a_{j_2} \cdots a_{j_m}$

So, we got that $t_{j_m} t_{j_{m-1}} \cdots t_{j_1} = b_{j_m} b_{j_{m-1}} \cdots b_{j_1}$. that is a match of P.

All in all, P has a match iff G is ambiguous.

Thus, the reduction from Post Correspondence Problem to $AMBIG_{CFG}$ works.

$AMBIG_{CFG}$ is undecidable.

5.24

Let A be the language $\{<M, x> \mid M$ is a TM and M doesn't accept $x\}$.
By reduction from A_{TM}, it can be checked that A is not Turing-recognizable.
First, we will reduce A to J.
Using reduction function $f(y) = 1y$
So, we got that y is in A if and only if $f(y)$ is in J.
As we know, the fuction f is computable, and A is not Turing-recognizable.
Thus, J is not Turing-recognizable.
After that, we will reduce $\overline{A_{TM}}$ to $\overline{J}$.
Using reduction function $g(y) = 0y$.
So, we got that y is in $\overline{A_{TM}}$ if and only if $g(y)$ is in $\overline{J}$.
Also, as we know that the function g is computable and $\overline{A_{TM}}$ is not Turing-recognizable.
Thus, $\overline{J}$ is not Turing-recognizable.

5.25

If there exist a Turing machine that accepts and halts on every input string of the language then that language is known as Decidable or Recursive. All the decidable language is also Turing-Acceptable. If the language of all yes instances to A is decidable then decision problem A is also decidable. Any Turing-recognizable works instead of co-Turing-recognizable language works (or vice versa).
For Example: A_{TM}. $(x \in A_{TM} \leq_m y \in \overline{A_{TM}})$

5.30

(c) $ALL_{TM} = \{<M> \mid M$ is a TM and $L(M) = \Sigma^*\}$

Let ALL_{TM} be the language such that ALL_{TM} is a description of Turing Machines.
It's non-trivial as some TM accepts Σ^*, and other do not have it.
It's satisfies the two conditions of rice's theorem.
Some TM's accept all possible string of an alphabet.
If some language is recognized by 2 Turing Machines, either both have descriptions in ALL_{TM} or neither will have.
So. using rice theorem prove. ALL_{TM} is undecidable.

6.13

Given for each $m > 1$. $Z_m = \{0, 1, 2, \cdots, m-1\}$ is a universe. $F_m = (Z_m, +, \times)$ be the model whose universe is Z_m. $+, \times$ are relations over modulo m.

So, we know that Z_m is finite.

Let Q_i are quantifiers. φ has no quanti ers.

A formula $\phi = Q_1 x_1 Q_2 x_2 \cdots Q_k x_k \varphi(x_1, x_2 \cdots x_n)$.

Define I_l for $0 \leq l \leq n$

$I_n(x_1 x_2 \cdots x_n) = \varphi(x_1 x_2 \cdots x_n)$

If Q_l is $\exists$. $I_{l-1}(x_1 x_2 \cdots x_{l-1}) = \bigvee_{i=0}^{m-1} I_i(x_1 x_2 \cdots x_{l-1}, i)$

If Q_l is $\forall$. $I_{l-1}(x_1 x_2 \cdots x_{l-1}) = \bigwedge_{i=0}^{m-1} I_i(x_1 x_2 \cdots x_{l-1}, i)$

$\Rightarrow$ We can simply annmerate all the possible values into formula and the formula is true.

By induction argument it is proved. The theory $Th(F_m)$ is decidable.