B62005Y-02 理论计算机科学基础	2017 年 5 月 3 日

Solutions to Homework # 4

主讲教师：杨光　　　　张远航 2015K8009929045

Sipser 5.1 Assume EQ_{CFG} is decidable and there exists a decider deciding it. That is, constructing a TM $M_{ALL_{\mathsf{CFG}}}$ to decide is possible as follows.

$M_{ALL_{\mathsf{CFG}}}$ = "On input CFG $\langle G\rangle$:

1. Derive a CFG G_{ALL} generating all possible strings.
2. Run $D_{EQ_{\mathsf{CFG}}}$ to decide whether $\langle G_{ALL}\rangle$ and $\langle G\rangle$ are equivalent.
3. Accept if it accepts; reject, otherwise."

However, ALL_{CFG} is undecidable, which contradicts the assumption that EQ_{CFG} is decidable.

Sipser 5.2 Since A_{CFG} is decidable, $M_{A_{\mathsf{CFG}}}$ can be used to test if a string $w \in G$, where G is a CFL. A Turing-recognizer $M_{\text{co-}EQ_{\mathsf{CFG}}}$ for the complement of EQ_{CFG} can be constructed in the following to show EQ_{CFG} is co-Turing-recognizable.

$M_{\text{co-}EQ_{\mathsf{CFG}}}$ = "On input CFGs $\langle G_1, G_2\rangle$

1. Enumerate strings with lengths in an ascending order.
2. For each string w enumerated:
3. Run $M_{A_{\mathsf{CFG}}}$ to decide if $w \in L(G_1)$ and if $w \in L(G_2)$.
4. Accept if either $w \in L(G_1)$ and $w \notin L(G_2)$ or $w \notin L(G_1)$ and $w \in L(G_2)$.
5. Continue otherwise."

Sipser 5.3 One possible match:

$$\left[\frac{\mathtt{ab}}{\mathtt{abab}}\right]\left[\frac{\mathtt{ab}}{\mathtt{abab}}\right]\left[\frac{\mathtt{aba}}{\mathtt{b}}\right]\left[\frac{\mathtt{b}}{\mathtt{a}}\right]\left[\frac{\mathtt{b}}{\mathtt{a}}\right]\left[\frac{\mathtt{aa}}{\mathtt{a}}\right]\left[\frac{\mathtt{aa}}{\mathtt{a}}\right]$$

Sipser 5.4 No. Consider the languages $A = \{\mathtt{0}^n\mathtt{1}^n : n \in \mathbb{N}\}$ and $B = \{\mathtt{1}\}$ over $\Sigma = \{\mathtt{0}, \mathtt{1}\}$. Define the function $f : \Sigma^* \to \Sigma^*$ by

$$f(w) = \begin{cases} \mathtt{1}, & \text{if } w \in A; \\ \mathtt{0}, & \text{otherwise.} \end{cases}$$

Note that f is a computable function and $w \in A$ if and only if $f(w) \in B$. Hence, $A \leq_m B$. However, A is not regular, while B is regular.

Sipser 6.1 Such a program is called a *quine*. Here is a program in C:

```
main(){char *s="main(){char *s=%c%s%c;printf(s,34,s,34);}";
printf(s,34,s,34);}
```

Sipser 6.2 Assume for sake of contradiction that there exists some infinite subset of MIN_{TM} that is Turing-Recognizable. Let this subset be S. Since S is Turing-recognizable, we have that there exists some enumerator E that enumerates S. Consider the following Turing machine M:

$M =$ "On input x:

1. Obtain own source code $\langle M \rangle$.

2. Use E to enumerate elements of S until you find a TM T output by E such that $|\langle M \rangle| < |\langle T \rangle|$.

3. Output $T(x)$."

We have that step 1 is possible by the recursion theorem. In step 2 we will eventually find such a TM T because S is infinite and hence its elements' lengths are not bounded. Hence, we have that the above Turing machine M effectively simulates T on all its inputs. i.e., M accepts, rejects and loops on exactly those inputs that are accepted, rejected and looped on respectively by T. Therefore M is equivalent to T. But we have that $|\langle M \rangle| < |\langle T \rangle|$. This is a contradiction as T is supposed to be minimal. Hence, MIN_{TM} has no infinite Turing recognizable subsets.