15-455: UCT **K. Sutner**

Assignment 6 Due: **March 21**, 2024.

1. Volatile Memory (40)

Background

Prof. Dr. Alois Wurzelbrunft is fascinated by Turing machines and likes to build them from discarded electronics. His current model, WMach, works pretty well, but the memory he uses for the machine tape is volatile: any symbol written in a cell will decay into a blank after a while. To safe the information, the tape head has to return to the cell and recharge it before it goes blank. At substantial cost, Wurzelbrunft can control the length of the decay, but, once the machine starts computing, the decay is fixed.

Wurzelbrunft suspects that these machines have a decidable Halting Problem (say, on empty tape). Hence, he would like to understand the complexity decision problem where the machine is encoded as some string, and the delay δ as the string 0^δ.

Problem:	**WMach Halting**
Instance:	A WMach machine M, a delay δ written in unary.
Question:	Does the machine halt on the empty tape?

By Halting we mean that the machine reaches a particular state q_H, it's fine if the computation continues beyond this point.

Task

A. Explain informally why WMach Halting is decidable.

B. Formalize the idea of a Turing machine with "volatile" tape.

C. Find a complexity class $\mathcal{C}$ so that WMach Recognition is $\mathcal{C}$-complete.

Comment

Think of the cell under the tape head as always being reenergized, even if the new symbol is the same as the old one. For simplicity, assume that the machines have to move right or left at each step. For part (C), make sure to establish both membership and hardness, something like $\mathcal{C} = \Delta_1$ will not work.

2. Small Space (30)

Background

Write $\mathsf{bin}_k(x) \in \mathbf{2}^\star$ for the binary expansion of $0 \le x < 2^k$, padded to k digits and LSD first (MSD first also works but is slightly clumsier). Define the languages

$$K = \{\, 0^n 1^n \mid n \ge 0 \,\} \subseteq \mathbf{2}^\star$$

$$L = \{\, C_k \mid k \ge 0 \,\} \subseteq \{0, 1, \#\}^\star \qquad \text{where}$$

$$C_k = \mathsf{bin}_k(0) \,\#\, \mathsf{bin}_k(1) \,\#\, \ldots \,\#\, \mathsf{bin}_k(2^k{-}1)$$

For example 00#01#10#11 is a string in L.

Recall that SPACE($o(\log\log n)$) is already SPACE(1), but that is as far as one can go: with $\log\log n$ space we can do something "useful" that a finite state machine cannot do.

Task

A. Show that the context-free language K is in $\mathbb{L}$.

B. Show that L is not regular.

C. Show that L is in SPACE($\log\log n$).

Comment For part (C) be clear about what exactly the size of the input string n is.

3. $\mathbb{NL}$ and Projections (30)

Background

Our definition of $\mathbb{NL}$ is based on nondeterministic Turing machines. As mentioned in class, there is an alternative way to define this class via projections, much in the spirit of our original definition of $\mathbb{NP}$:

$$x \in A \iff \exists\, w \in \mathbf{2}^{p(|x|)} \left(\mathcal{M} \text{ accepts } (w, x) \right)$$

Here $\mathcal{M}$ is a deterministic log-space checker, p a polynomial and the witness w is given to $\mathcal{M}$ on a separate, read-once input tape. The actual instance x is still on a standard read-only tape. For the sake of brevity, let's call such a machine tame.

Task

A. Show that one can obtain SAT in this way if we drop the read-once condition.

B. Show that every language recognized by a tame checker is in $\mathbb{NL}$.

C. Show that every language in $\mathbb{NL}$ can be accepted by a tame checker.

Comment Explain your coding method for part (A) in some detail, so the logarithmic space bound becomes clear.