15-455: UCT	**K. Sutner**
Assignment 6	Due: **March 25**, 2022.

1. Small Space (20)

Background

Write $\mathsf{bin}(x) \in \mathbf{2}^\star$ for the binary expansion of $x \in \mathbb{N}$. For simplicity assume LSD first (though the claim also holds for MSD first). Define the languages

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

$$L = \{\, \mathsf{bin}(0) \,\#\, \mathsf{bin}(1) \,\#\, \ldots \,\#\, \mathsf{bin}(n) \mid n \geq 0 \,\} \subseteq \{0, 1, \#\}^\star$$

For example 0#1#01#11#001#101 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($\log\log n$).

2. Regular Expression Equivalence (20)

Background

Two regular expressions α and β are equivalent if they denote the same language: $\mathcal{L}(\alpha) = \mathcal{L}(\beta)$. Some equivalences are trivial, say, $\alpha_1 + \alpha_2$ is equivalent to $\alpha_2 + \alpha_1$, but in general the algebra of regular expressions is fairly complicated (thanks to the Kleene star operation), and it is difficult to check equivalence with algebraic methods. For example,

$$(\alpha^k)^*\big(\varepsilon + \alpha + \alpha^2 + \ldots + \alpha^{k-1}\big) = \alpha^\star$$

which is pretty wild from an algebraic perspective.

At any rate, one would like to understand the complexity of the following decision problem:

Problem:	**Regular Expression Equivalence (REE)**
Instance:	Two regular expressions α and β.
Question:	Are α and β equivalent?

Task

A. Describe a real algorithm to solve REE.

B. What is the time/space complexity of your algorithm?

C. Explain how to implement an equivalence test in PSPACE.

Comment

REE turns out to be PSPACE-complete ($\mathbb{NP}$-complete for a single-letter alphabet), but we won't go there.

3. More Tilings (30)

Background

Recall the tiling problem from the previous homework: we discussed the pointed square tiling (PST) problem: can a $n \times n$ square can be tiled by a given set of tiles (with SW and NE corners fixed)? This problem turns out to be $\mathbb{NP}$-complete.

Define the pointed rectangular tiling (PRT) problem as follows: given a "width" n, is there a "height" m so that the $n \times m$ rectangle can be tiled (again assuming a given SW and NE tile). So the input is the tile set, the 2 corner tiles, and 0^n; by contrast, m is not part of the input.

Task

A. Show that the pointed rectangle tiling problem is in PSPACE.

B. Show that the pointed rectangle tiling problem is PSPACE-hard.

Comment

There is no need to repeat all the details of the old PST construction, just explain what the essential differences are in this version of the problem.

4. CSL Emptiness (30)

Background

Membership in a CSL L is trivially decidable, but even the CSL-Emptiness problem (is $\mathcal{L}(G) = \emptyset$?) is already undecidable. This is slightly surprising, since one might be tempted to think that the argument for context-free grammars could somehow be lifted to the context-sensitive scenario.

Let $\mathcal{M}$ be some arbitrary Turing machine.

Task

A. Find a convenient (for part B) way to express computations of $\mathcal{M}$ as strings over some alphabet.

B. Show that the language of all accepting computations of $\mathcal{M}$ is context-sensitive.

C. Conclude that CSL-Emptiness is undecidable.