Tutorial 6

Exercise 1 (compulsory)

We know that the problem ALL_{CFG} (does a given context-free grammar recognize all strings from Σ^*?) is undecidable.

Consider the problem $EQ_{CFG} \overset{\text{def}}{=} \{\langle G_1, G_2\rangle \mid G_1 \text{ and } G_2 \text{ are CFGs such that } L(G_1) = L(G_2)\ \}$.

- By reduction from ALL_{CFG} prove that EQ_{CFG} is undecidable.
- Prove that EQ_{CFG} is co-recognizable.
- Can EQ_{CFG} be also recognizable?

Solution:

- First we show that EQ_{CFG} is undecidable by reduction from ALL_{CFG}.

  1. By contradiction. Assume that we have a decider R for EQ_{CFG}.
  2. Using R we construct a decider S for ALL_{CFG} as follows:

```
S = " On input $\langle G\rangle$ where $G$ is a CFG over the set of terminals $\Sigma$:
      1. Let $G_1$ be a context-free grammar with start nonterminal $S$
         which for every $a \in \Sigma$ contains the rule $S \to aS$
         plus it has the rule $S \to \epsilon$. (Note that $L(G_1) = \Sigma^*$.)
      2. Run $R$ on $\langle G, G_1\rangle$.
      3. If $R$ accepted then $\underline{S \text{ accepts}}$.
         If $R$ rejected then $\underline{S \text{ rejects}}$."
```

  3. Clearly, S is a decider for ALL_{CFG} because checking the totality of G is reduced to checking of equivalence between G and G_1 where $L(G_1) = \Sigma^*$. But we know that S cannot exist, so R does not exist either.
  4. Conclusion: EQ_{CFG} is undecidable.

- Now we shall argue that EQ_{CFG} is co-recognizable. Consider the following TM M (which is a recognizer).

```
M = "On input $\langle G_1, G_2\rangle$ where $G_1$ and $G_2$ are CFG over the alphabet $\Sigma$:
    1. Repeatedly generate in lexicographical order all strings from $\Sigma^*$.
    2. For each such string $w$ check whether $w \in L(G_1)$ and $w \notin L(G_2)$,
       or $w \notin L(G_1)$ and $w \in L(G_2)$.
    3. If one of the checks is positive then $\underline{M \text{ accepts}}$, else take the next string $w$ and goto step 2.
```

  Observe that the checks in step 2. can be done algorithmically because A_{CFG} is a decidable language (see Lecture 3 or Theorem 4.7 on page 172). Hence M will only loop if $L(G_1) = L(G_2)$. On the other hand if the languages are not equal, the machine M will sooner or later find a string which is in one language but not in the other one and accept. Hence M accepts if and only if $L(G_1) \neq L(G_2)$ and it is so a recognizer for $\overline{EQ_{CFG}}$. By definition it follows that EQ_{CFG} is co-recognizable.

- No, EQ_{CFG} cannot be recognizable because we just argued that $\overline{EQ_{CFG}}$ is recognizable and hence if EQ_{CFG} was recognizable too, the language would be necessarily decidable (by Theorem 4.22 on page 183), but this is not the case.

Exercise 2 (compulsory)

Consider the following instance P of Post correspondence problem.

$$P = \{ [\frac{ab}{abab}], [\frac{b}{a}], [\frac{aba}{b}], [\frac{aa}{a}] \}$$

Find a match of P (a nonempty sequence of indices between 1 and 4).

Solution:
The following sequence of indices is a match: $4, 4, 2, 1$.

Exercise 3 (compulsory)

Consider the problem *silly Post correspondence problem* (SPCP). An instance P of SPCP is

$$P = \{ [\frac{t_1}{b_1}], [\frac{t_2}{b_2}], \dots, [\frac{t_k}{b_k}] \}$$

where $t_i, b_i \in \Sigma^*$ and moreover $|t_i| = |b_i|$ for all i, $1 \leq i \leq k$. In other words in each pair the top string has the same length as the bottom string. The question is whether P contains a match (like in standard PCP).

- Is the problem SPCP decidable or undecidable? Give a precise proof of your claim.

Solution:
The SPCP problem is decidable. It follows from the following claim.

Claim: A given SPCP instance has a match if and only if there is a domino $\left[\frac{t_i}{b_i}\right]$ such that $t_i = b_i$.

Proof of the claim:
"$\Rightarrow$": If a SPCP instance has a match it has to start with some domino. Because the length of the top and the bottom string is the same in all dominos, the first domino in the match must surely have the same top and bottom string.
"$\Leftarrow$": If there is a domino with the same top and bottom string then this single domino forms a trivial match of SPCP.

Finally, checking whether there is a domino with the same top and bottom string is easily decidable by examining the SPCP instance.

Exercise 4 (compulsory)

Consider the problem *binary Post correspondence problem* (BPCP). An instance P of BPCP is

$$P = \{ [\frac{t_1}{b_1}], [\frac{t_2}{b_2}], \dots, [\frac{t_k}{b_k}] \}$$

where $t_i, b_i \in \{0,1\}^*$ for all i, $1 \leq i \leq k$.

In other words BPCP instance is like PCP instance, we only restrict the alphabet for the top and bottom strings to consist only of two symbols.

- Is the problem BPCP decidable or undecidable? Give a precise proof of your claim.

Hint: Consider an instance of standard PCP over some general alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$. Is it possible to construct an instance of BPCP which has a match if and only if the original instance has a match?

Solution:
The problem BPCP is undecidable. We prove this by demonstrating a reduction from PCP to BPCP.

1. By contradiction assume that we have a decider R for BPCP.
2. We will construct a decider S for PCP as follow. The decider S will on a given input of the form $\langle P \rangle$ where P is a PCP instance over some alphabet $\Sigma = \{a_1, a_2, \ldots, a_n\}$ construct an instance P' of BPCP such that P has a match if and only if P' has a match. To produce P' from P, for all i, $1 \leq i \leq n$, we simply replace every occurrence of the symbol a_i in P with the sequence of symbols $0\underbrace{11\ldots1}_{i \text{ times}}$.

   Hence for example the domino $\left[\frac{a_1a_3}{a_2a_4a_1}\right]$ will be changed to $\left[\frac{010111}{0110111101}\right]$. The idea is that the symbol 0 works as a separator and the actual letters from Σ are represented in a unary way by a corresponding sequence of 1's of a certain length. It is now clear that this new P' has a match if and only if P has a match.

   As we assume that we have a decider R for BPCP, we can run R on $\langle P' \rangle$ and it will tell us whether P' has a match or no. The decider S for PCP will simply overtake this answer from R.
3. Hence we constructed a decider S for PCP but we know that it cannot exist. This means that R, the decider for BPCP, does not exist either.
4. Conclusion: BPCP is undecidable.

Exercise 5 (optional)

Consider the problem whether two given context-free grammars have a nonempty intersection of the languages they generate.

- Describe the decision problem as a language $INTERSECTION_{CFG}$.
- Prove that $INTERSECTION_{CFG}$ is undecidable by reduction from PCP.

Hint: For a given PCP instance

$$P = \{[\frac{t_1}{b_1}], [\frac{t_2}{b_2}], \ldots, [\frac{t_k}{b_k}]\}$$

you might find useful to consider the following grammars G_1 and G_2 (where $i_1, i_2, \ldots, i_k$ are new terminal symbols).

$$\begin{aligned} G_1 : \quad & S_1 \to t_1S_1i_1 \mid \ldots \mid t_kS_1i_k \mid t_1i_1 \mid \ldots \mid t_ki_k \\ G_2 : \quad & S_2 \to b_1S_2i_1 \mid \ldots \mid b_kS_2i_k \mid b_1i_1 \mid \ldots \mid b_ki_k \end{aligned}$$

Exercise 6 (optional)

Show that any Turing machine that is allowed to use only the first $2|w|$ tape cells when run on an input w is equivalent to LBA (which can use only $|w|$ of the first tape cells).

Exercise 7 (optional)

Problem 5.17 on page 216 and Exercise 5.8 on page 215.