Theoretical Computer Science - Bridging Course  
Winter Term 2016  
Exercises for the Additional Tutorial

Formal Languages - Pumping Lemma

Show that the following languages are not regular in case of (a),(b) and not context-free in case of(c). Use the respective Pumping lemma.

(a) \{a^m b^m a^m \mid m,n \in \mathbb{N}\}
(b) \{a^m b^n \mid m \neq n\}
(c) \{a^n ba^{2n}ba^{3n} \mid n \in \mathbb{N}\}

Solution

(a) Let \(p \in \mathbb{N}\) be the pumping length. Pick \(s = a^p b^p a^p\), which is in \(L\) and longer than \(p\). Let \(xyz = s\) be a partition for which \(|xy| \leq p\) and \(y \geq 1\). Then \(y = a^k\) with \(1 \leq k \leq p\). We pump with \(i = 2\). Then \(xy^2z = a^{p+k}b^p a^p \notin L\). This shows that the pumping lemma does not hold for \(L\) which shows that \(L\) can not be regular.

(b) Call \(L := \{a^m b^n \mid m \neq n\}\) our language from the exercise. Then \(\{a^k b^k \mid k \geq 0\} = L \cap a^*b^*\). Assume that \(L\) is regular. Then \(\overline{L}\) is also regular due to the closure properties of regular languages. Thus also \(\overline{L} \cap a^*b^* = \{a^k b^k \mid k \geq 0\}\) is regular, which is a contradiction since we know \(\{a^k b^k \mid k \geq 0\}\) is not regular from the lecture (shown similarly to (a)).

Decidability - Halting Problem

(a) Consider the language \(\{s\}\) containing the single string \(s\) which is defined as follows

\[
s = \begin{cases} 
1, & \text{if } \mathcal{P} = \mathcal{NP} \\
0, & \text{else.}
\end{cases}
\]

Is \(\{s\}\) decidable?

(b) Is the Halting problem undecidable? Is it semi-decidable?

(c) Let \(L_d := \{(M,s) \mid \text{Turing machine } M \text{ rejects input string } s\}\). Is \(L_d\) decidable?

**Hint:** Assume that \(L_d\) is decidable, i.e. it has a decider \(M_d\). Construct a Turing machine that incorporates \(M_d\) and derive a paradox (i.e. a contradiction).
Complexity Theory - Polynomial Reductions

(a) Show that Connected := \{\langle G \rangle \mid G \text{ is a simple, undirected, connected graph} \} \in \mathcal{P}.

(b) Consider sets of 'items' of the form I := \{i_1, \ldots, i_n\} and two integers V, W \in \mathbb{Z}. Each item i_j = (v_j, w_j) is a tuple consisting of a value v_j \in \mathbb{Z} and a weight w_j \in \mathbb{Z} of that item and we are looking for a selection of items such their value is bigger than V but their weight smaller than W. Accordingly we define the following problem

Knapsack := \{\langle I, V, W \rangle \mid \text{set of items } I \text{ has a subset } J \subseteq I, \text{ s.t. } \sum_{j \in J} v_j \geq V, \sum_{j \in J} w_j \leq W \text{ where } V, W \in \mathbb{Z} \}.

Show that Knapsack \in \mathcal{NP}.

(c) Consider the following, known \mathcal{NP}-hard problem

\text{SubsetSum} := \{\langle S, X \rangle \mid \text{set } S \text{ of integers has a subset } T \subseteq S, \text{ s.t. } \sum_{t \in T} t = X \text{ where } X \in \mathbb{Z} \}.

Show that \text{SubsetSum} \leq_p \text{Knapsack}.

Solution

(a) –

(b) –

(c) We show that Knapsack can be used to solve the \mathcal{NP}-hard SubsetSum (SubsetSum \leq_p Knapsack). Knowing that Knapsack \in \mathcal{NP} from part (b) we derive that SubsetSum \in \mathcal{NP}C.

So let’s do the polynomial reduction SubsetSum \leq_p Knapsack. We have to give a mapping f that is computable in poly. time and that maps an instance \langle S, X \rangle of SubsetSum onto an instance \langle I, V, W \rangle =: f(\langle S, X \rangle) of Knapsack such that

\langle S, X \rangle \in \text{SubsetSum} \iff f(\langle S, X \rangle) \in \text{Knapsack}.

For \langle S, X \rangle we define I := \{\langle s, s \rangle \mid s \in S\} (which means that weight and value are equal for every item in I) and define V := W := X (the weight and value requirements are equal to X). The mapping f can be computed in polynomial time, because we construct the item-set I in \mathcal{O}(n) for n := |S|, and computing V, W requires only constant time.

It remains to prove the equivalency stated above. Let \langle S, X \rangle \in \text{SubsetSum}. This means there is a subset T \subseteq S such that \sum_{t \in T} t = X. Thus \sum_{t \in T} t \leq X = W and \sum_{t \in T} t \geq X = V. Then let J := \{\langle t, t \rangle \mid t \in T\} \subseteq I. We see that this selection J of the items I fulfills our requirement. Therefore f(\langle S, X \rangle) = \langle I, V, W \rangle \in \text{Knapsack}.

Conversely let \langle S, X \rangle \notin \text{SubsetSum}. Thus for all subsets T \subseteq S it is \sum_{t \in T} t \neq X! This means that one of the inequalities \sum_{t \in T} t \leq X = W or \sum_{t \in T} t \geq X = V must be violated. Therefore there can be no subset J \subseteq I that satisfies both of these conditions. Hence \langle I, V, W \rangle \notin \text{Knapsack}. 

2