

**COMPUTATIONAL COMPLEXITY THEORY (COMS 30126)**  
**EXERCISE SHEET 1**

(1) TMs: read Sipser §3.1 and look through one example there (say example 3.4).

(2) Suppose that the definition of TMs is modified to allow the machine to “stay put” i.e. the transitions are now of the form

$$(q, x) \rightarrow (q', x', d) \quad \text{where } d \text{ is L or R or S}$$

and S means that the head stays over the same tape square. Show that such more general machines are no more powerful for recognising or deciding languages than those in our original definition i.e. given any machine  $\mathcal{M}_{new}$  of the new sort, show that there's a machine  $\mathcal{M}_{old}$  of the original sort such that on any input  $\omega$ ,  $\mathcal{M}_{new}$  and  $\mathcal{M}_{old}$  behave in the same way (i.e. both accept, both reject or both don't halt).

(3) (i) Let  $\mathcal{L}_1$  and  $\mathcal{L}_2$  be Turing decidable languages. Show that the union  $\mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2 = \{w : w \in \mathcal{L}_1 \text{ or } w \in \mathcal{L}_2\}$  is also decidable.

(ii) Let  $\mathcal{L}_1$  and  $\mathcal{L}_2$  be Turing recognisable languages. Show that the union  $\mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2$  is also recognisable.

(In the answers it suffices to give suitably high level descriptions of how various machines operate.)

(4) Let  $\mathcal{A}$  be the language containing only the single string  $w$  where

$$w = \begin{cases} 0 & \text{if God does not exist} \\ 1 & \text{if God does exist.} \end{cases}$$

Is  $\mathcal{A}$  decidable? Why or why not? (Note: the answer does not depend on your religious convictions.)

(5) (i) Determine whether each of the following is true or false:

$$2n = O(n), n^2 = O(n), n^2 = O(n \log^2 n), 2^{2^n} = O(2^{2^n}), 15 = O(1)$$

$$n = o(2n), 2n = o(n^2), 1 = o(1/n), 15 = o(1).$$

(ii) The notation  $f(n) = 2^{O(g(n))}$  means  $f(n) \leq 2^{cg(n)}$  for some (fixed)  $c$  and all  $n >$  some  $n_0$ . Show that  $3^n = 2^{O(n)}$  (Is  $3^n = O(2^n)$ ?). Also show that if  $f(n) = 2^{O(\log n)}$  then  $f(n) = O(n^c)$  for some  $c$ .

...continued

(6) Closure properties of P.

(i) For any language  $\mathcal{L} \subseteq \Sigma^*$  its complement  $\mathcal{L}'$  is the language  $\mathcal{L}' = \Sigma^* - \mathcal{L} = \{w : w \notin \mathcal{L}\}$ . If A is any class of languages, co-A is the class of all complements of languages in A. Show that co-P = P.

(ii) Show that P is closed under unions (i.e. if  $\mathcal{L}_1, \mathcal{L}_2$  are in P then so is  $\mathcal{L}_1 \cup \mathcal{L}_2$ ).

Similarly one can show that P is closed under intersections, concatenations  $\mathcal{L}_1 \circ \mathcal{L}_2 = \{w : w = w_1w_2 \text{ for some } w_1 \in \mathcal{L}_1 \text{ and } w_2 \in \mathcal{L}_2\}$ , and the star operation  $\mathcal{L}^* = \{w : w = w_1w_2 \dots w_n \text{ for some } w_1, w_2, \dots, w_n \in \mathcal{L} \text{ and some } n \in \mathbb{N}\}$ .

Prove the result for concatenation. Optional: prove the result for the star operation. (Hint: on input  $w = a_1a_2 \dots a_n$  with  $a_i \in \Sigma^*$  build a table indicating for each  $i < j$  whether  $a_i \dots a_j \in \mathcal{L}$  or not).

(7) Consider the following two tasks:

FACTORING: given an integer  $N$  find  $1 < k < N$  such that  $k$  divides  $N$ . If  $N$  is prime, return 1.

Decision problem X: Given a pair of integers  $(N, M)$  decide if  $N$  has a factor in the range  $1 < k < M$ .

(i) Show that a poly time algorithm for X implies a poly time algorithm for FACTORING. (Hint: consider a suitable binary search).

(ii) Conversely, show that a poly time algorithm for FACTORING implies a poly time algorithm for X. (harder; Hint: note that the number of prime factors of  $N$  is always less than  $\log N$ ; and  $N$  has a factor  $< M$  if and only if the smallest prime factor of  $N$  is  $< M$ ).