11.5.12

(a)

Show that it is undecidable to answer the following problem called **RP**-ACCEPTANCE: Given a precise non-deterministic Turing machine (PNTM) M, is it true that for all inputs either all computations reject or at least half of them accept?

Reduction from the complement of HALTING. Assume a deterministic Turing machine M' and an input x. Build the machine M such that, for input y, (i) M first builds a yard stick $\sqcap^{|y|}$ on an extra string, (ii) then prints x (which is a constant for M) on the input string and (iii) finally non-deterministically initiates four deterministic computations of length $|y|$ (the yard stick is used) such that (a) three of them do nothing, halting on "no" after $|y|$ steps, and (b) one of them executes M' on x for $|y|$ steps and halts on "yes" if M' halted within time $|y|$ and on "no" otherwise. Now if M' halts on x then there is an input y of sufficient length such that $\frac{1}{4}$ of the $2^{q(|y|)+|x|+2+|y|}$ computations of M on accept and $\frac{3}{4}$ of them do not (where $q(|y|)$ is the time taken in computing the yard stick). In that case, M does not fulfill the **RP** accepting condition. On the other hand, if M' does not halt on x, then all computations of M always reject for all inputs y and M fulfills the **RP** accepting condition. Therefore, if the **RP**-ACCEPTANCE were decidable, then the complement of HALTING would be, meaning that **RP**-ACCEPTANCE is undecidable.

(b)

Show that it is undecidable to answer the following problem called **BPP**-ACCEPTANCE: Given a precise non-deterministic Turing machine (PNTM) M, is it true that for all inputs either at least $\frac{3}{4}$ of the computations reject or at least $\frac{3}{4}$ of them accept?

Reduction from the complement of HALTING. Assume a deterministic Turing machine M' and an input x. Build machine M such that, for input y, (i) M first builds a yard stick $\sqcap^{|y|}$ on an extra string, (ii) then prints x on the input string and (iii) finally non-deterministically initiates two deterministic computations of length $|y|$ (the yard stick is used) such that (a) one of them does nothing, halting on "no" after $|y|$ steps, and (b) one of them executes M' on x for $|y|$ steps and halts on "yes" if M' halted within time $|y|$ and on "no" otherwise. Now if M' halts on x then there is an input y of sufficient length such that $\frac{1}{2}$ of the computations of M accept. In that case, M does not fulfill the **BPP** accepting condition. On the other hand, if M' does not halt on x, then all two computations of M always reject for all inputs y and M fulfills the **BPP** accepting condition. Therefore, if **BPP**-ACCEPTANCE were decidable, then the complement of HALTING would be, meaning that **BPP**-ACCEPTANCE is undecidable.

(c)

Show that it is undecidable to answer the following problem called **TFNP**-ACCEPTANCE: Given a precise non-deterministic Turing machine (PNTM) M, is it true that for all inputs at least one computation accepts?

Reduction from the complement of HALTING. Assume a deterministic Turing machine M' and an input x. Build machine M such that, for input y, (i) M first builds a yard stick $\sqcap^{|y|}$ on an extra string, (ii) then prints x on the input string and (iii) executes M' on x for $|y|$ steps and rejects if M' halted within time $|y|$ and accepts otherwise. Now if M' halts on x then there is an input y of sufficient length such that M rejects on all computations. In that case, the answer to **TFNP**-ACCEPTANCE on M is "no". On the other hand, if M' does not halt on x, then all computations of M always accept for all inputs y and the answer to **TFNP**-ACCEPTANCE on M is "yes". Therefore, if **TFNP**-ACCEPTANCE were decidable, then the complement of HALTING would be, meaning that **TFNP**-ACCEPTANCE is undecidable.

(d)

Show that it is undecidable to answer the following problem called **NP** $\cap$ **coNP**-ACCEPTANCE: Given two precise non-deterministic Turing machines (PNTM), M_1 and M_2, is it true that for all inputs either the first has an accepting computation, or the second does, but never both?

Reduction from the complement of HALTING. Assume a deterministic Turing machine M' and an input x. Build machine M_1 such that, for input y, (i) M_1 first builds a yard stick $\sqcap^{|y|}$ on an extra string, (ii) then prints x on the input string and (iii) executes M' on x for $|y|$ steps and accepts if M' halted within time $|y|$ and rejects otherwise. Build machine M_2 such that, for input y, (i) M_2 first builds a yard stick $\sqcap^{|y|}$ on an extra string, (ii) then prints x on the input string and (iii) does nothing for $|y|$ steps and accepts. Now if M' halts on x then there is an input y of sufficient length such that M_1 and M_2 both accept. In that case, the answer to **NP** $\cap$ **coNP**-ACCEPTANCE on M_1 and M_2 is “no”. On the other hand, if M' does not halt on x, the M_1 always rejects and M_2 always accepts and and the answer to **NP** $\cap$ **coNP**-ACCEPTANCE is “yes”. Therefore, if **NP** $\cap$ **coNP**-ACCEPTANCE were decidable, then the complement of HALTING would be, meaning that**NP** $\cap$ **coNP**-ACCEPTANCE is undecidable.

11.5.13

Déjà vu: A class $\mathcal{C}$ is *closed under reductions* if it holds that whenever a language L reduces to a language $L' \in \mathcal{C}$, then $L \in \mathcal{C}$.

The problem is to show that the classes **RP**, **BPP** and **PP** are closed under reductions.

Proof by hand-waving. Note that each class in question can also perform deterministic computation as explained in the book. Now assume that a reduction from a language L exists to a language $L' \in \mathcal{C}$ where $\mathcal{C} \in \{\mathbf{RP}, \mathbf{BPP}, \mathbf{PP}\}$. Clearly $L \in \mathcal{C}$ since it is decided by the following precise machine M fulfilling the accepting condition required for $\mathcal{C}$: M first performs the (log-space and therefore polynomial length) deterministic reduction function (which was assumed to exists) on the input x and then executes the $\mathcal{C}$-machine M' on the transformed input, deciding L' (one has to exist since $L' \in \mathcal{C}$). Clearly M is a $\mathcal{C}$-machine and decides L by the definition of reductions.

11.5.15

The problem is to show that **PP** is closed under complement.

Complement

A class $\mathcal{C}$ of languages is closed under complement if it holds that: for all languages $L \in \mathcal{C}$, the complement $\overline{L}$ of L is in $\mathcal{C}$.

Consider a precise non-deterministic polynomial-time Turing machine M that **PP**-decides L. Let $p(n)$ be the precise polynomial bound of M. Build a precise non-deterministic polynomial-time Turing machine M' as follows. On input x, M' first computes the yard stick $p(|x|)$ on an extra string. M' then non-deterministically does one of the following: (i) it executes M on x (which takes time precisely $p(|x|)$) and rejects if M accepts and vice versa or (ii) it executes a deterministic computation of length $p(|x|)$ by using the yard stick and accepts on $2^{p(|x|)-1}+1$ computations and rejects on $2^{p(|x|)-1}-1$ computations. Now the length of each computation of M' is $q(|x|)+1+p(|x|)$, where $q(|x|)$ is the polynomial amount of time consumed in computing the yard stick. Let $k = 2^{q(|x|)}$.

Now if $x \in L$, then at most $k(2^{p(|x|)-1} - 1 + 2^{p(|x|)-1} + 1) = k(2^{p(|x|)})$ of $k(2^{p(|x|)+1})$ computations accept x, i.e., at most half of the computations of M' accept x. On the other hand, if $x \notin L$, then at least $k(2^{p(|x|)-1}+2^{p(|x|)-1}+1) = k(2^{p(|x|)} + 1)$ of $k(2^{p(|x|)+1})$ computations accept x, i.e., more than half of the computations of M' accept x. Therefore M' **PP**-decides $\overline{L}$ and **PP** is closed under complement.