Selected solutions week 11 - IN2080

The following are solutions to some of the weekly tasks which we discuss in the group sessions, which are not present in the book. Note that we haven't formally defined arithmetic operations for Big O terms, so some of the analyses below are more so an informal one rather than formal.

7.6

Let M and N decide the languages A and B respectively in polynomial time, i.e. $A, B \in P$.

Union

Let the following machine decide the language $A \cup B$ in polynomial time:

$G =$ "On input w:

1. Simulate M on w. If it accepts; *accept*.
2. Simulate N on w. If it accepts; *accept*, otherwise $reject$."

This machine decides the union of the two languages, as both M and N are deciders, and a string is in the union iff either machine accepts.

Both M and N run in $O(n^k)$ time for some k. As such, the runtime of G is $O(n^k) + O(n^k) = O(n^k)$, and thus $L(G) \in P$.

Concatenation

Let the following machine decide the language AB in polynomial time:

$G' =$ "On input w:

1. For each split of $w = w_1 w_2$:
2. Simulate M on w_1, and N on w_2. If both accept; *accept*.
3. If no split of w was accepted; $reject$."

We decide the language by trying each possible split of the input, running the first machine on the first part of the input, and the second machine on the second part, accepting only when both machines accept.

Both M and N run in polynomial time $O(n^k)$, so step 2 runs in $2 \cdot O(n^k) = O(n^k)$. This is repeated for each split of w of which there are $n + 2 = O(n)$ many. As such, the machine runs in $O(n) \cdot O(n^k) = O(n^{k+1})$, and $L(G') \in P$.

Complement

This should be obvious; we just simulate M on w and accept iff it rejects, and vice versa. Since M runs in polynomial time, the complement is in P.

7.7

Let M and N decide the languages A and B respectively in non-deterministic polynomial time, i.e. $A, B \in NP$.

Union

Let the following machine decide the language $A \cup B$ in non-deterministic polynomial time:

$G =$ "On input w:

1. Non-deterministically choose to simulate M or N on w. *Accept* if the simulation accepts, *reject* otherwise."

The non-deterministic choice is $O(1)$, and simulating either machine is done in $O(n^k)$ time, so the whole machine is run in non-deterministic polynomial time, and $L(G) \in NP$.

Here, a verifier could take one of the machine's non-deterministic choices as the certificate, and simulate the steps on w. As the longest possible path from the root to a leaf on the computation tree of a non-deterministic polynomial time TM is, by definition, $O(n^k)$, this simulation would also be done in polynomial time (we've not considered the runtime of the simulation itself, which involves more tape head movement than the machine we're simulating, but it should be intuitive that this will not lead to an exponential growth in steps).

Concatenation

Let the following machine decide the language AB in non-deterministic polynomial time:

$G =$ "On input w:

1. Non-deterministically choose a split of $w = w_1 w_2$

2. Simulate M on w_1, and N on w_2. If both accept; *accept*, otherwise; $reject$."

Both M and N run in non-deterministic polynomial time, so step 2 is on $O(n^k)$. This applies to a non-deterministic choice as well, so the machine runs in $O(n^k)$.

One could also make a verifier, where the certificate would be the non-deterministic choices of M and N, alongside the split of w. Following the same argumentation from the previous task, this will also be deterministically polynomial.

7.8

The algorithm is the following:

M = "On input $\langle G \rangle$, the encoding of a graph G:

1. Select the first node of G and mark it.
2. Repeat the following stage until no new nodes are marked:
3. For each node in G, mark it if it is attached by an edge to a node that is already marked.
4. Scan all the nodes of G to determine whether they all are marked. If they are, *accept* ; otherwise, $reject$."

Step 1 and 4 is only done once, and both in $O(n)$ steps. Step 3 is repeated at most $O(n)$ times, as there are finitely many nodes, and at least one new node is marked for the loop to continue. Step 3 loops over each node and each edge for a node. As such, step 3 takes $O(n) \cdot O(n) = O(n^2)$ (note that we could be more fine-grained and instead say $O(|V|) \cdot O(|E|)$, where V are the nodes of G and E the edges. This, however, doesn't change what we're trying to show – that the language is in P – so we'll not be that detailed). This results in the runtime $O(n) + O(n) \cdot O(n^2) + O(n) = O(n^3)$. The machine decides $CONNECTED$ in polynomial time, and thus $CONNECTED \in P$.