Quarter-term Formative Exercise Sheet for Weeks 1-3

ID: 1803086

February 25, 2021

1. Show that

$$(A \in \mathbf{L} \cap B \in \mathbf{L}) \implies C \in \mathbf{L}$$

   We know that both w and w' are in $\mathbf{L}$ as their respective supersets are known to be in $\mathbf{L}$

   Therefore we can simply split any element $x \in C$ into w and w' and decide them separately in $\mathbf{L}$, as $\mathbf{L}$ is closed under addition, the space usage of deciding w and w' is also in $\mathbf{L}$ this is the case as $\log(w) + \log(w') \equiv \log(ww')$.

2. (a) Given that we know addition to be computable in polytime. If we can reduce multiplication to addition we can show it too is computable in polynomial time.

   We can think of multiplication as repeated addition. $a \times b$ can be thought of as a added to a $b - 1$ times.

   Our machine M can be constructed with 2 tapes as follows.

   Say we encode a and b as a zeros and b zeros respectively, separated by a 1, we write this to our input tape.

   We perform addition on these two numbers by copying each 0 on the input tape to the output tape, we skip over the 1 and we terminate when we reach a blank. At the end of the process we have $a + b$ 0s on the output tape.

   For each 0 before the break (1) we copy the value of b from the input tape to the output tape and move the head to the right. When we reach the break (1), we terminate. $a \times b$ is present on the output tape.

   As such we can show the complexity of multiplication to be the same as addition, multiplied by a constant factor $(b - 1)$ and so can be ignored.

(b) To show that $\overline{\textbf{PRIME}}$ is in **NP** we must show that a polytime machine exists to check polynomially sized certificates of $\overline{\textbf{PRIME}}$. I.e. a machine that can compute:

$$\exists a, b \in \mathbb{N}.(a \times b = n) \cap (a \neq n) \cap (b \neq n)$$

A (potential certificate of $\overline{\textbf{PRIME}}$) is a pair of a, b values that satisfies the above.

Our machine M to check this can be thought as having 3 tapes. The input tape takes a, b and n in the same format as the machine from the previous part (0s separated by 1s). The result of $a \times b$ is then computed in the same was as above and copied to the first work tape.

We now start the process of checking $a \times b = n$, $a \neq n$ and $b \neq n$.

To check the value on the first work tape is n we move the input tape head to the start of n, the first value after the second 1 and the work tape head to the first 0. We then move both heads to the right each time they both read 0, if at any point they do not match: write 0 to the output tape and return, upon reaching blank on both tapes, move to the next check.

To check $a \neq n$ move both heads to the first 0. Move both heads to the right each time they are the same, when the input head reads a 1 if the work head reads blank write 0 to the output tape and halt. Otherwise, if the input head reads 1 and the work head reads 0, move to the next check.

To check $b \neq n$ perform the same process as before but start the input head at the first 0 after the 1st 1.

We have shown $a \times b$ to be computable in polytime in part a. the comparisons ($a \times b = n$, $a \neq n$, $b \neq n$) are also trivially computable in polynomial time. Therefore, the requisite machine exists so $\overline{\textbf{PRIME}}$ is in **NP**

3. (a) In the case where hospitals make offers to students, h_1 is matched with s_3. This is the case as the initial conflict between h_3 and h_4 over s_1 is resolved by allocating $h_4 s_1$ as s_1 prefers h_4 and allocating h_3 its second (free) choice s_2

   (b) In the case where students make offers to hospitals, s_4 is matched with h_2. This is the case as when s_2 and s_3 compete for h_1, s_3 *wins* as h_1 prefers s_3. s_2 is then freed. s_2 then competes with s_4 for h_2 and again *loses* as h_2 prefers s_4. s_2 is then assigned h_3 as it is free.

4. Show that `INDEPENDENT SET` $\leq_{\mathbf{P}}$ `VERTEX COVER`

   Given a vertex cover `Vert` of $G = (V, E)$, we can deduce the independent set of G in polytime. The set `Vert` contains the set of vertices that *cover* all edges in G, therefore, any vertices in G **not** in the vertex cover form an independent set as no edges exist between them. This reduction can easily be done in polynomial time by calculating $V \backslash$`Vert`