Algorithms & Complexity (Extended) Summative Midterm Controlled Assessment

1803086

March 10, 2021

1. (a) Given that $L_1, L_2 \in \textbf{coNP}$, show that $L_1 \cap L_2 \in \textbf{coNP}$ and $L_1 \cup L_2 \in \textbf{coNP}$

   We already know that if $L_1, L_2 \in \textbf{coNP}$ then $\bar{L_1}, \bar{L_2} \in \textbf{NP}$ as well as $\bar{L_1} \cap \bar{L_2} \in \textbf{NP}$ and $\bar{L_1} \cup \bar{L_2} \in \textbf{NP}$.

   If we let $L_3 = L_1 \cap L_2$ and $L_4 = L_1 \cup L_2$, we therefore know that $\bar{L_3}, \bar{L_4} \in \textbf{NP}$, so therefore, their compliments L_3 and L_4 are in **coNP**

   If $L_3, L_4 \in \textbf{coNP}$ then we can therefore say that $L_1 \cap L_2 \in \textbf{coNP}$ and $L_1 \cup L_2 \in \textbf{coNP}$

   (b) To modify our machine M to decide the compliment of L, we use the simple fact that any string decided by M to **not** be in L must then be in $\bar{L}$. Our machine M' can therefore be constructed to run M on an input x and write its result to our final (non-output) work tape. We then write $1-$ the result to our output tape and halt. Though this would use an extra cell compared with M, the time complexity would still be in the same class as inverting and writing a bit is $O(1)$.

   Alternatively, one could edit M such that if it finds x to be in L is writes 0 to the output tape and halts and otherwise writes 1 to the output tape and halts.

   (c) From (b) we have a method for deciding the compliment of a language with a polytime $(O(t(n)))$ machine. Therefore by definition of **P** as $\bigcup_{k\geq 1} \textbf{DTIME}(n^k)$ All languages in **coNP** must be decidable in polynomial time and therefore are in **P**, meaning, $\textbf{P} = \textbf{coNP}$.

2. (a) The preference lists are as follows:

	1st	2nd	3rd	4th
h_1	s_1	s_2	s_4	s_3
h_2	s_2	s_3	s_1	s_4
h_3	s_4	s_1	s_3	s_2
h_4	s_3	s_1	s_4	s_2

	1st	2nd	3rd	4th
s_1	h_1	h_2	h_4	h_3
s_2	h_2	h_3	h_1	h_4
s_3	h_4	h_1	h_3	h_2
s_4	h_3	h_1	h_4	h_2

The above preference lists satisfy all given properties. This is the case as for each hospital, any properties are enforced by placing them as low down in preference as possible. I.e. the allocation would only be made if there was no way to avoid it. We know that Gale-Shapely always matches `Best`(h) to h for each hospital h, and looking at the hospital preference lists, there is no source of conflict in the first position, meaning the required properties are un -visited at the end of each respective list.

(b) As it turns out, my preference lists for part (a) satisfy this requirement. In both tables, there is no conflict in the first column. Each first column specifies the same set of matchings so therefore Gale shapely when run either way on these tables returns this (same) matching.

3. (a) In the case where $N = 9$, the greedy algorithm would first do a long run of 6, followed by 3 days of short runs (1), as the remaining distance is shorter than the medium run (4), to make 9 totalling 4 days. The optimal number of days would in this case be 3, two medium runs followed by a single short run.

(b) If we were to generate a times table list for each of the run lengths up to a given N (or as close to it as we can). We can then find the lowest index in any of these three lists to contain N. If this lands in the d-list we subtract d from N, store d on a stack and repeat until we reach 0, at which point the stack contains `OPT`$[N]$

(c) In this case all run lengths are divisible by the lower lengths. The result of this is that, at any time if $N > 4$, run length l will be the optimal choice as the alternatives are simply diving a single run into multiple runs across multiple days. At any point the largest possible run to be chosen is the optimal choice. There is no way of diving a number N into smaller parts by dividing by a smaller number. There is no $N, N \geq 4$ such that $\frac{N}{1} < \frac{N}{2} < \frac{N}{4}$.