CS605 - Tutorial (Emailed questions & sample papers)
30 April 2025

Part I -- students' emailed questions

Pumping Lemma for Regular and Non-regular languages

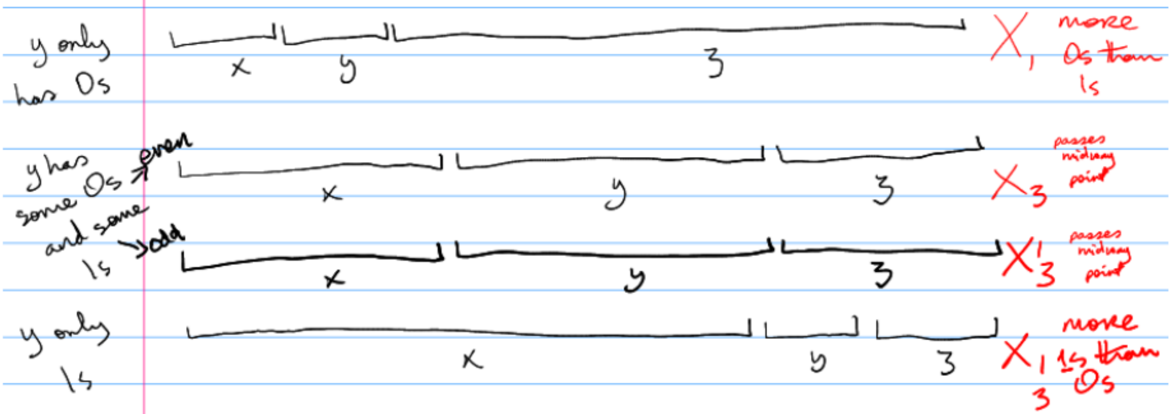
Doubt 1. I believe the reasoning for the 2nd partition here, i.e., y has some 0s and 1s(even) length is incomplete
why? --> because y can be even and not have the same number of 0s and 1s e.g., y=0111 in this if we pump y we will have more 1s than 0s (or vice versa if we modify the y)
---> the given reasoning will only hold for when we have equal no. Of 0s and 1s in y
[please correct me if I am wrong]

$C = \{w : w \in \{0,1\}^*, w \text{ has an equal number of } 0\text{s and } 1\text{s}\}$

Prove C is not regular.

Proof: We will prove this by contradiction using the pumping lemma.
Assume C is regular. Let p be the pumping length given by the pumping lemma. Let $w = 0^p 1^p$.
According to the pumping lemma, w can be divided into three pieces $w = xyz$ satisfying three conditions.
These are all the ways that w can be divided:

$0_1 0_2 0_3 \ldots\ldots 0_p 1_1 1_2 1_3 \ldots\ldots 1_p$



There is no way to pump this word. A contradiction because the pumping lemma states it should be possible. Therefore this proves C is not regular.

Doubt 2. In most cases after a certain point the rule 3 breaks everywhere, so in the exam it will be fair to not look for any other rule since rule 3 already breaks. Is that correct?

Doubt 3. How to know that the word that we have chosen for the pumping lemma is a right word and satisfies that the language is not regular? (what if there is a different word for might be completely off the track here professor)

Response to student's emailed question on previous page:

Doubt 1. Yes, I understand your reasoning. Your way is fine. Note, you will still need two cases, so it is not shorter than my way.
- equal 0s and 1s
- not equal 0s and 1s

My way is not incorrect, because I am being very conservative: with each case I am only concluding it breaks a rule if it breaks that rule in all cases. I try to explain this as follows.

In the case of "even length y", it will break sometimes and not break other times so I say conservatively "is not guaranteed to break rule 1" but it is "guaranteed to break rule 3".

In the case of "odd length y", it is "guaranteed to break rule 1" and it is "guaranteed to break rule 3".

I hope this explains that we are both correct.

Doubt 2. Yes, you are 100% correct, but it requires a little bit more sophisticated thought so for beginners I do the proofs the long way: listing all of the partitions first, and then proving that each one is not possible. If you do it a shorter way in the exam, you can still get full marks if it is correct, but if it is incorrect and you don't fully explain the shortcut you are taking, I may not understand exactly what you are doing and may not be able to give you marks for it.

Doubt 3. You will never know, until your proof works. You may choose badly a few times, and have to start your whole proof again each time. However, will more and more practice, it becomes quicker and quicker to think of a suitable word. My experience of students who get full marks is that sometimes they will make the right choice first time, but almost always students get it right on their second try; very rarely does someone need a third try.

Part II -- overview of past exam questions

All past exam papers on the module webpage 2016-2024 were looked through, question by question, and it was explained which past questions are valid revision questions for CS605 for the 2024-2025 academic year.

Part III -- overview of past exam questions

Verbal questions from students -- several excellent questions, but they were not minuted.

Part IV -- selected questions from CS605 sample papers 2025A and 2025B

See pages that follow.

[10 marks]

3 Prove that the problem associated with language L3 defined below is undecidable. You are given that HALT = {<*M*, *w*>: *M* is a Turing machine and *M* halts on *w*} is undecidable. Use the template provided to perform a mapping reduction. You must give your answer on this exam question sheet.

The language L3 is defined as
L3 = {<*M*>: *M* is Java program that takes no input and when *M* runs it does not throw (raise) exception number 21}.

Note, in the template below, some blanks have a small subscript number. Blanks with the same subscript number must have the same value. Blank *N* = "..." is worth 10 marks, and each other blank scores –1 if incorrect.

[10 marks]

4 In the case of L3 defined earlier, and its complement $\overline{\text{L3}}$, prove whether each language is Turing-recognisable or not Turing-recognisable.

You can refer to your previous proof that L3 is undecidable in your answer. Note, you are not told whether L3 is Turing-recognisable or not. You must give your answer on this exam question sheet.

$\overline{\text{L3}}$ = {<*M*>: *M* is Java program that takes no input and when *M* runs it ~~does not~~ throws (raise) exception number 21}.

We will prove that $\overline{L3}$ is T-r. TM N recognises $\overline{L3}$.

N = "On input <M>:
1. Run M, keeping track of any exceptions thrown.
2. If M throws exception 21, accept."

Since TM N recognises $\overline{L3}$ this proves $\overline{L3}$ is T-r.

Since $\overline{L3}$ is T-r and in Q3 we proved L3 is undecidable, this proves that L3 is not T-r, (because otherwise L3 would be decidable).

[10 marks]

6 Prove that each of these problems is in $\mathcal{NP}$. You must use this exam question sheet for your proofs.

(a) L6A = {<A> : $A = \{x_1, ..., x_m\}$ is a set of natural numbers and two (possibly overlapping) subsets $Y,Z \subset A$ exist such that each element of A is in at least one of the two subsets, and the sum of the elements in Y equals the sum of the elements in Z}. [5 marks]

As an example,
<{1, 3, 4, 7}> ∈ L6A because we can have subsets Y = {1, 3, 4} and Z = {1, 7}.

(b) L6B = {<M, n, W> : M is a set of finite automata, n is an integer where $n > 0$, W is a set of n words over {a, b}*, each $w \in W$ has length exactly n, and at least one $w \in W$ is accepted by each machine in M}. [5 marks]

(b) We will prove this language is in NP by constructing a polynomial time TM N to recognize it.

N = "On input <M, n, W>:
1. For each FA F in M: [X]
2. accepted = False [1]
3. For each u in W: [n]
4. Run F on u. [n]
5. If F accepts, accepted = True [1]
6. If not accepted, reject. [1]
7. Accept." [1]

Let X be the number of machines in M.

The Running time is $X(1+n(n+1)+1)+1$

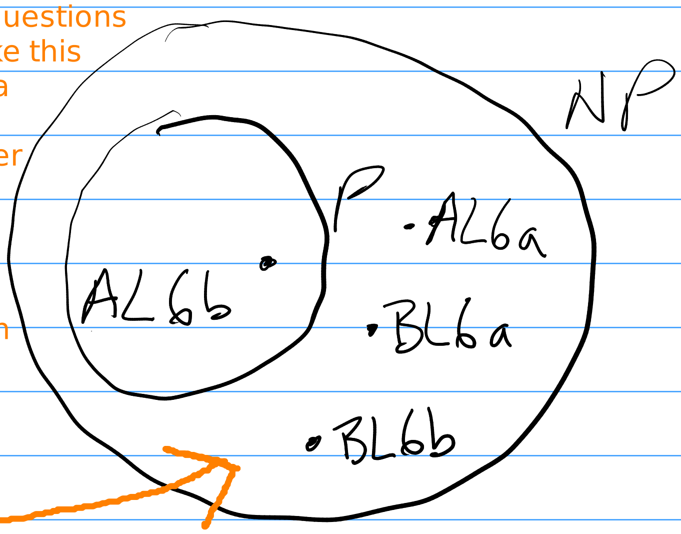
$$X(n^2+n+2)+1$$

$$Xn^2+Xn+2X+1$$

$$= O(Xn^2)$$

Since TM N recognises language L6b in polynomial time, this proves that L6b is in P. Since $P \subseteq NP$ this also proves that L6b is in NP.

Note, very few NP questions can be answered like this question that uses a polynomial-time recogniser; the other three NP questions in the sample papers require a polynomial-time verifier, as shown in the next example.



[10 marks]

6 Prove that each of these problems is in $\mathcal{NP}$. You must use this exam question sheet for your proofs.

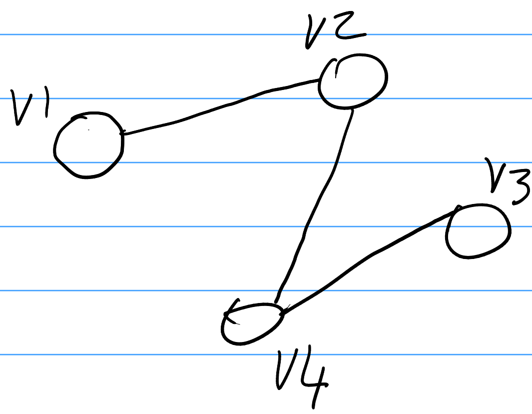
(a) L6A = {<G, *k*>: G = (*V*, *E*) is an undirected graph consisting of a set *V* of students in a class and set *E* of pairs of students in *V* that are friends with each other on social media website CARA, and a subset $S \subseteq V$ of size *k* exists where all students in *S* are CARA friends with each other} [5 marks]

(b) Backstory: In a security company, a set *V* of employees are suspected of leaking company secrets. A set *E* exists of pairs of employees (*a*, *b*), where $a,b \in V$, that have sent encrypted private messages to each other using company phones. The boss believes the employees in *V* may have been discussing company secrets. The boss wants to know the minimum number of company phones that need to be accessed to view all private messages sent between employees in *V*. [5 marks]

L6B = {<*V*, *E*, *k*>: *V* is a set of employees' phones and *E* is a set of pairs of employees' phones that have been used to send encrypted private messages between them. There exists a subset $S \subseteq V$ of phones of size $|S| = k$ such that each element of *E* contains one of the phones in *S*}

(b)

Aside:



$k = 2$

We will prove this lang. is in NP by constructing a polynomial time verifier M for L6B. Let certificate c be the subset of phones.

M = "On input $\langle V, E, k, c \rangle$:

1. Check $c \subseteq V$. [Mk]
2. Check $|c| \leq k$. [k]
3. Check each e in E contains at least one vertex in c. [2Nk]
4. If all checks pass, accept." [3]

Aside

Step 3 could be described in more detail as:

For each e in E: [N]

Check if e.[0] is in c [k]

Check if e[1] is in c [k]

Let $|V| = M$, $|E| = N$

The running time is $Mk + k + 2Nk + 3$

$= O(Mk + Nk)$

Since TM M verifies LGB in polynomial time, this proves that LGB is in NP.