COMP 314 Final Exam Spring 2023 Solution

Qu 1

L is regular if & only if its reverse L^R is regular. So we will show L^R is not regular.

Assume L^R is regular and argue for a contradiction.

L^R is an infinite language, so the pumping lemma applies. Let N be the pumping cutoff. We may assume $N \geq 5$.

Consider the string

$$T^N A^5 G^N C^5$$

The pumping lemma tells us that some nonempty substring in the first N characters can be pumped. This must consist of T^K for some $K \geq 1$. Pumping twice gives

$$T^{N+K} A^5 G^N C^5$$

which is not in L^R, contradicting the pumping lemma. $\square$

Qn 2

We show YesOnEmpty $\leq_T$ Increment5OnSome,
(YoE) (I5OS)

which will show I5OS is uncomputable because YoE is uncomputable.

Let P be an instance of YoE. Create a new program P′ that operates on input I′ as follows:

- set $v = P(\varepsilon)$ [ie. simulate P on empty input]
- if $v =$ "yes" and I′ represents an integer M,
  return M+5
  else return "no"

By construction, $P'(M) = M+5$ for all integers M if P is a positive instance of YoE, and $P'(M) =$ "no" for all inputs if P is a negative instance of YoE.

Hence, we can solve YoE by passing P′ to I5OS, completing the reduction.

OR: Use Rice's Thm

Qu 3

YesOnEmpty (YoE) is recognizable but not decidable.
Given a positive instance P, we can recognize P by simulating P(ε), which is guaranteed to terminate with output "yes".
Given a negative instance, the simulation might not terminate, which suggests we cannot always decide negative instance. As we know, this can be proved rigorously by a reduction from YesOnString.

Qu 4

line 6: negP2Instance = "0 ; 0 0 ; 0 0" (because weights and thresholds must be > 0).

line 13: weights.append (5) any value will work.

14: L1 = L
H1 = H
L2 = 5
H2 = 5

Qu 5a

A computer is a person who performs calculations.

Qu 5b

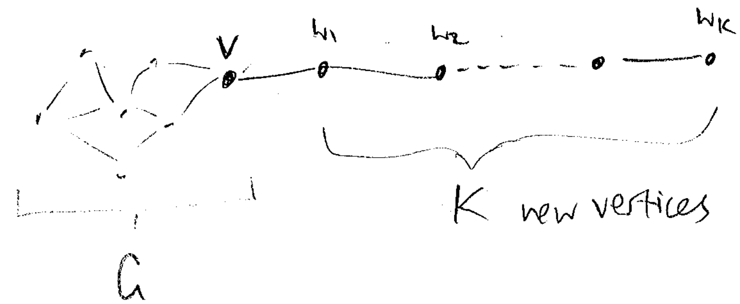
Computable can refer to a number that could be calculated by a human performing calculations OR a number that can be output by a Turing machine. In fact, Turing argues that these different definitions actual define the same set of 'computable' numbers.

Qu 6

a) We can verify a positive instance G using a hint that consists of a Hamilton cycle with K repeats. It is clear that the hint can be checked in polynomial time, by checking it is a cycle and counting the number of times each vertex is visited.

b) We show $UHC \le_p$ KRepeatsHamCycle (KRHC), which will show NP-completeness as we know $KRHC \in NP$ and UHC is NP-complete.

Let G be an instance of UHC.
Pick any vertex of G - say vertex v. Add K new vertices $w_1, w_2, \dots w_K$ to G in a chain connected to v.



The resulting graph, G', has a Hamilton cycle with K repeats if & only if G has a Hamilton cycle. In detail:

- if G has a Ham cycle, we insert the sequence $v w_1 w_2 \dots w_K w_{K-1} \dots w_2 w_1 v$ where v occurs, obtaining a Ham cycle with K repeats.
- if G' has a Ham cycle with K repeats, it must include $v w_1 w_2 \dots w_K w_{K-1} \dots w_2 w_1 v$. We replace this sequence with just 'v', obtaining a Ham cycle in G.

Finally note the conversion runs in poly time since only K vertices are added. 16mk/30

Qn 7 $(\neg x_1 \vee \neg x_2 \vee x_5 \vee x_7) \wedge (x_1 \vee x_3 \vee \neg x_5 \vee x_7 \vee x_8)$

$(\neg x_1 \vee \neg x_2 \vee d_1) \wedge (\neg d_1 \vee x_5 \vee x_7) \wedge (x_1 \vee x_3 \vee \neg x_5 \vee d_2) \wedge (\neg d_2 \vee x_7 \vee x_8)$

$(\neg x_1 \vee \neg x_2 \vee d_1) \wedge (\neg d_1 \vee x_5 \vee x_7) \wedge (x_1 \vee x_3 \vee d_3) \wedge (\neg d_3 \vee \neg x_5 \vee d_2) \wedge (\neg d_2 \vee x_7 \vee x_8)$

Qn 8 We know from the textbook that a problem is unrecognizable if its complement is recognizable but undecidable.

The complement is SomeLowerCase (SLC), defined as follows.

Given program P, SLC solution is "yes" if P can produce a lowercase letter, and "no" otherwise. SLC is recognizable, since we can simulate P nondeterministically on all possible inputs and return 'yes' as soon as a lowercase letter is output.

But SLC is undecidable. We show this by proving $YoE \leq_T SLC$.

Given instance P of YoE, construct P' as:

- set $v = P(\varepsilon)$
- if $v =$ 'yes' return 'a'
  else return 'A'

[or use Rice's Thm here]

Since P' can produce a lowercase output if & only if P is a positive instance, the reduction is complete.