CMPS130: Midterm 2

Name (Please print): KEVIN WANG

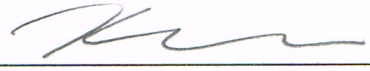
UCSC email (Please print): KWANG43@GMAIL.COM

Note:

1. Print this out double-sided, and write your answers in the space provided. After that, scan it and push a copy to your git repo. Do this before the deadline given below.
2. **Turn your exam in during class on Tue, Feb 26. If you have a DRC accommodation, turn it in by 5PM Wed, Feb 27. You can slide your exam copy under my office door, E2-347A.**
3. You are not allowed to talk to anyone else about the questions. Please do the exam yourself.
4. You are not allowed to use the internet. Please avoid searching for anything related to this course, until your submit the midterm.
5. You can refer to any books, class notes, or written material.
6. You may post clarification questions to Piazza, but please do not answer anyone else's question. The instructors will answer these queries.
7. I do not expect formal proofs, but I require some precision in language. If you are (say) modifying a DFA to accept another language, precisely state the modifications. Give some reasoning as to why the modification works. If we don't understand your solution, we cannot give you full credit.
8. You can cite any result explained in class, without providing a proof. Just precisely state what property/theorem from class you are using.

Please sign below to indicate your agreement with the following honor code.

Honor code: I promise not to cheat on this exam. I will neither give nor receive any unauthorized assistance. I promise not to share information about this exam with anyone who may be taking it at a different time. I have not been told anything about the exam by someone who has already taken it.

Signature:  **Date:** 2/26/2019

Part 1: Short questions

- There are 6 short questions, each of 1 point.
- You should not need more 5 sentences for an answer. Please do not be too verbose. I am not looking for a detailed formal proof. Just give a succinct explanation.
- If you are trying to disprove a statement about languages, give a clear counterexample. Also, explain why the counterexample is valid.
- For multiple choice questions or True-False questions, clearly mark your answer. Then, give a short explanation of your answer. No credit without an explanation. Randomly guessing will not help.

Q1 (1 point): True or False. If a language $\mathcal{L}$ is undecidable, then $\mathcal{L}$ is not RE.
If you answer "False", you must provide a counterexample.
Answer: FALSE

L_{HALT} is undecidable

Let $L_{Halt} = \{\langle M, x\rangle \mid M \text{ halts on } x\}$ where $U_{TM}(\langle M, x\rangle)$ simulates M on x (and reproduces output Acc/Rej). (lec11)

We know L_{HALT} is undecideable by contradiction / at a TM D given $\langle D\rangle$ as input. (lec11, Turing 36)

L_{HALT} is RE

We also know that the U_{TM} accepts L_{HALT} as a language (but cannot decide it), such that $L_u \subset L_{HALT}$. Thus, the language L_{HALT} is recursively enumerable (RE). ✱

Therefore, a language L that is undecideable can still be recursively enumerable. □

✱ Also lec11.pdf states: "L_{HALT} and L_ω are recursively enumerable".

Q2, (1 point): True or False. There is a language $\mathcal{L}$ that is co-RE but not RE.
If you answer "True", you must provide an example.
Answer: TRUE

L_{HALT} is co-RE

Let $L = \overline{L_{Halt}}$. We know L_{HALT} is RE (lec11), thus $\overline{L}$ is RE, and L is co-RE (by def).

$\overline{L_{HALT}}$ is RE

Assume L is RE, such that $\overline{L_{HALT}}$ is RE.

L_{HALT} is decidable, contradiction

Then L_{HALT} is co-RE, which implies it is decideable. This is a contradiction as L_{HALT} is undecideable (lec 11).

Therefore, $L = \overline{L_{HALT}}$ is a language that is co-RE but not RE. □

✱ Reference lec12: "$\overline{L_{HALT}}$ is NOT RE ... $\overline{L_{HALT}}$ is co-RE"

Q3, (1 point): Prove that there is no printer for $\overline{\mathcal{L}_u}$. (Recall that $\mathcal{L}_u$ is the universal language.)
Answer:

is RE + decidable: Let L_u be the universal language. We know that L_u is undecidable and RE (Hopcroft Thm 9.6)

$\overline{L_u}$ has printer, is RE: Assume $\overline{L_u}$ has a printer. Then, $\overline{L_u}$ must be RE. (lec12: "L has a printer $\iff$ L is RE).

is decidable (contradiction): This implies that L_u is co-RE and is decidable. (lec 11)

Thus, by contradiction, there is no printer for $\overline{L_u}$. □

Q4, (1 point): Give an example of undecidable languages $\mathcal{L}, \mathcal{M}$, such that $\mathcal{L} \cup \mathcal{M}$ is decidable. Precisely state the languages $\mathcal{L}$ and $\mathcal{M}$.
Answer:

L, M are undecidable: Let U be an undecidable set. We define L, M as such:

$$L = \{0x \mid x \in \Sigma^*\} \cup \{1x \mid x \in U\}$$

$$M = \{1x \mid x \in \Sigma^*\} \cup \{0x \mid x \in U\}$$

L, M, are both undecidable (due to U)

L ∪ M is decidable: However $L \cup M \Rightarrow \{0x \mid x \in \Sigma^*\} \cup \{1x \mid x \in \Sigma^*\}$

$$\Rightarrow \Sigma^*$$

Thus, L ∪ M is decidable.

Q5, (1 point): Prove that the following is *false*: $\overline{\mathcal{L}_{Halt}} \leq_m \mathcal{L}_{Halt}$.
Answer:

$\overline{L_{Halt}}$ is co-RE + undecidable { We know that $\overline{L_{HALT}}$ is co-RE (because L_{HALT} is RE). We also know both $\overline{L_{HALT}}$ and L_{HALT} are undecidable.

f true { Assume $\overline{L_{Halt}} \leq_m L_{Halt}$ is true.

$\overline{L_{Halt}}$ is cidable. tradiction { If $\overline{L_{Halt}}$ is co-RE, then L_{Halt} must also be co-RE (lec 13). Then, $\overline{L_{Halt}}$ would be decidable which is a contradiction

Therefore, "$\overline{L_{Halt}} \leq_m L_{Halt}$" is FALSE. □

Q6, (1 point): True or False. Undecidability is closed under complementation.
If you answer "False", you must provide a counterexample.
Answer:

cidability sed under mplement { We know that decidability is closed under complement (lec12), where L is decidable iff $\overline{L}$ is decidable.

is decidable { Let M be an undecidable language.

$\overline{M}$ is cidable { Assume undecidability is NOT closed under complement, such that $\overline{M}$ is decidable.

is cidable ontradiction { Then, because decidability is closed under complement M must be decidable, which is a contradiction.

Thus, $\overline{M}$ must be undecidable proving that undecidability is closed under complementation. □

Part 2: Longer Questions

- There are 2 questions in this section, each worth 2 points.
- Please keep your answer in the page designated for that question.
- Each question specifies a structure for your solution. **Please follow it.** It will make it easier for you to get partial credit.
- I do not expect fully rigorous proofs. But when I ask for "an argument", I expect a logically coherent explanation (typically 3-4 sentences).

Q7 (2 points): Prove that the language of binary strings with an equal number of 0s and 1s is not regular, but is a CFL.

Your answer should have four distinct parts that are clearly demarcated.

- Part 1 (1 point): a proof that the language is not regular.
- Part 2 (0.5 points): an explicit PDA or a CFG for the language.
- Part 3 (0.25 points): an argument that every string in the language is accepted by the PDA/CFG.
- Part 4 (0.25 points): an argument that every string accepted by the PDA/CFG is in the language.

Answer: Let $L = \{w \mid w$ has an equal # of 0's and 1's$\}$.

I. Assume L is regular. Then it must satisfy the pumping lemma

Let n be the number of states in L.
Let $w = 0^n 1^n$, such that $|w| = 2n \geq n$.
Let $w = xyz$, where by the pumping lemma, $|xy| \leq n$.
Thus, x and y consists of only 0's where $y \neq \epsilon$.
Let $k = 0$. If $xy^k z = xz$ is in L, then L is regular (pumping lemma). However, xz has n 1's (all 1's are in z) and has at most (n-1) 0's (we removed at least one 0 when removing y; remember $y \neq \epsilon$). Thus xz is not a part of L, contradicting our assumption of regularity. □

I. Let C be a CFG, as defined:

$V = \{S, X, Y\}$
$\Sigma = \{0, 1\}$
$P = \{S \rightarrow SOS1S \mid S1SOS\}$
$S = \epsilon$

Generally, the strings defined by C are made up of a 0 and 1 (order not specific), separated by a substring S which has an equal number of 0's and 1's. The string can also be ϵ.

Answer to Q7, continued:

III. Every string w in L is accepted by CFG C.

Prf. The empty string with n=0 1's and 0's is accepted by C where $S \longrightarrow \varepsilon$. (Base Case)

Assume a string $w \in L$ with $n>0$ 1's and 0's is accepted by C. (Induction Hypothesis)

We note that w can become three substrings $x, y, z \in L$ with i,j,k 0's and 1's where $i+j+k=n$. (Induction Step)
(Obvious possibility is have a substring be w, and rest be ε)

Then, by C's production rule, a string w' with n+1 0's and 1's is also accepted by C.

Thus, $\forall w \in L \longrightarrow w \in L(C)$. □

IV. Every string w accepted by CFG C is in L.

Prf. The only word w that can be generated in $k=0$ steps is ε, which is in L (has n=0 1's and 0's) (Base Cas

Assume for every $S \overset{*}{\Longrightarrow} w$ with $k \geq 0$ steps, w is in L. (Induction Hypothesis)

Let $S \overset{*}{\Longrightarrow} w$ have k+1 steps. Because it is not the start step, we look at the production rules.

$$P = \{S \longrightarrow S0S1S \mid S1S0S\}$$

We know that any S from a step before k+1 is in L and thus has an equal number of 0's and 1's. The production step k+1 then adds adds the same number (1) of 0's and 1's to the strings.

Therefore, the number of 0's and 1's remain equal for any w in L(C).

Thus, $\forall w \in L(C) \longrightarrow w \in L$ □

Q8 (2 points): Let us define a new kind of machine, called counter DFAs (CDFA). This is a DFA that also has access to a special register, called a *counter*. The counter stores a non-negative (unbounded) integer. The transition function also depends on whether the counter is zero or non-zero. Furthermore, each transition can increment or decrement the counter (if the counter is zero, a decrement does not change the value).

Formally, the transition function is of the form $\delta : Q \times \Sigma \times \{\text{zero, non-zero}\} \to Q \times \{\text{inc, dec}\}$. In other words, at each "step", the CDFA consumes an input symbol σ, and makes a transition depending on σ, the current state, and whether the counter is zero. The CDFA transitions to a new state and increments or decrements the counter. Acceptance is identical to DFAs, through reaching a set of final states.

Show: if $\mathcal{L}$ is the language of a CDFA, then $\mathcal{L}$ is a CFL.

Your answer should have the following parts, clearly demarcated.

- Main idea (0.75 points): in 2-3 sentences, explain the main idea behind your construction.
- Construction (1.25 points): given a CDFA, show how to construct an equivalent CFG/PDA.

Starting with a formal description of the CDFA, you need to mathematically describe the CFG/PDA (e.g. in case of a PDA, clearly write out the states, symbols, transition function, etc.). If you construct a PDA, you will get partial credit for every aspect you get correct (so partial credit for the right state space, the right transition, etc.).

Answer: Main Idea A CDFA is basically a DFA that modifies and keeps track of a counter. Because we can implement the counter with a stack, a CDFA can be described by a PDA (DFA with a stack). while a CDFA increments or decrements the counter, the PDA push or pops from the stack. While a CDFA makes a transition depending on whether the counter is zero, the PDA makes a transition depending on whether the stack is empty.

Construction Let CDFA $C=(Q, \Sigma, C, \delta, q_0, F)$ be a generic DFA with a non-negative integer counter C, and transition fu

$$\delta: Q \times \Sigma \times \{\text{zero, non-zero}\} \longrightarrow Q \times \{\text{inc, dec}\}$$

$$\delta(q, \sigma, c) = \big((q', c+1), (q', c-1), (q', c)\big)$$

Using C, we can construct PDA $P=(Q, \Sigma, \Gamma, \delta, q_0, \$, F)$

Q is the same $\Gamma = \{a, \$\}$ \$ is the start symbol on stack

Σ is the same q_0 is the same F is the same

$\delta: Q \times \Sigma \cup \{\epsilon\} \times \Gamma \longrightarrow Q \times (\Gamma^* \cup \{\epsilon\})$

$$\delta(q, \sigma, X) = \big((q', \$), (q', X)\big)$$

$$\delta(q, \sigma, \$) = \big((q', \$), (q', X)\big)$$

Answer to Q8, continued:

Description of PDA P: Assume some symbol σ and a zero counter, when given to CDFA C, results in a transition from q to q', and increases the counter. Then σ and a \$ being read from the stack, results in a transition from q to q' and pushes a marker X to the stack. (Similar description for other cases: decreasing counter = popping from stack, decreasing from a 0 counter = no change to empty stack)

If L is the language of CDFA C, then L is a CFL.

Prf. By construction, we know that PDA P works exactly like a CDFA as they share the base DFA and the counter (zero, non-zero) is replicated by the stack (empty, non-empty). As such, $L(C) = L(P)$.

We know that the languages defined by PDA's are exactly the context-free languages (CFL's) (Hopcroft 6.3). Thus, if $L = L(C)$, then L is a context free language. □