NAME: ____________________________

COMPSCI 250
Introduction to Computation
Solutions to Final Exam Spring 2022

D. A. M. Barrington and K.Doney

10 May 2022

DIRECTIONS:

- Answer the problems on the exam pages.
- There are five problems on pages 2-8, some with multiple parts, for 120 total points plus 10 extra credit. Probable scale is somewhere around A=105, C=70, but will be determined after we grade the exam.
- Page 9 contains useful definitions and is given to you separately – do not put answers on it!
- If you need extra space use the back of a page.
- No books, notes, calculators, or collaboration.
- In case of a numerical answer, an arithmetic expression like "$2^{17} - 4$" need not be reduced to a single integer.

Last month Blaze and Rhondda were invited to an Easter Egg Hunt at the home of their neighbor Clover. Their neighbor Kiké also joined them. Clover's family placed 23 plastic eggs in the yard, and each dog was given credit for each egg that they first found. For any of the four dogs, let $e(d)$ be the (natural) number of eggs found by dog d. Of the four dogs, Clover and Rhondda are Goldens, and the other two not. Rhondda and Blaze are female, and the other two not. The predicate $G(x)$ means "x is a Golden" and $F(y)$ means "y is female".

Question 1 (15): Translate the five statements as indicated:

- (a, 3) (to symbols) (Statement I) It is not the case that every dog found at least one egg.

  $\neg\forall d : e(d) > 0$ *or* $\exists d : e(d) = 0$

- (b, 3) (to English) (Statement II) $\forall d : (e(d) \neq 5) \rightarrow (\neg F(d) \vee G(d))$

  If any dog found a number of eggs other than five, then either it was not female or it was a Golden (or both).

- (c, 3) (to symbols) (Statement III) Rhondda found exactly twice as many eggs as were found by some non-female dog.

  $\exists d : \neg F(d) \wedge (e(r) = 2e(d))$

- (d, 3) (to English) (Statement IV) $\forall x : \forall y : (G(x) \wedge \neg G(y)) \rightarrow (e(x) > e(y))$

  Given any two dogs, if the first is a Golden and the second is not, then the first found more eggs than the second. Alternatively, every Golden found more eggs than any non-Golden.

- (e, 3) (to symbols) (Statement V) Some female dog found exactly five eggs.
  $\exists d : F(d) \wedge (e(d) = 5)$

Question 2 (15): Using the definitions and statements from Question 1, determine exactly how the 23 eggs were distributed among the four dogs. Prove your answers, and indicate where you have used any quantifier rules.

Since Rhondda found an even number of eggs (by III), she could not be the female with exactly five eggs (Instantiation on V), so it must be Blaze that found five. (It also follows from the contrapositive of Statement II that any female non-Golden must have found five.) Since Blaze is not a Golden, the two Goldens must each have found more than five by Specification on IV. By Instantiation on I, there must be a dog with no eggs, and this cannot be either a Golden or Blaze, so it must be Kiké. Rhondda found twice as many eggs as some non-female dog (Instantiation on III), and this dog cannot be Kiké, so it must be Clover. The total number of eggs found is thus $e(r) + e(c) + e(b) + e(k) = 2e(c) + e(c) + 5 + 0 = 3e(c) + 5$, *from which we may conclude that* $e(c) = 6$ *and therefore* $e(r) = 12$.

Question 3 (20+10): Let $\Sigma = \{a, b, c\}$. Let the language X be the set of all strings over Σ that never have a double letter (that is, with no consecutive appearances of the same letter). Let Y be the language of the regular expression $(abcab + abc)^*$.

- (a, 10) Let $f(n)$, for any natural n, be the number of strings of length n in X. Prove, for all *positive* naturals n, that $f(n) = 3 \times 2^{n-1}$. (**Hint:** Use ordinary induction on all positive naturals. Remember that your argument must refer to strings, not just numbers.)

  Base case: This must be $n = 1$ since we are proving this over all positive naturals. We need to show that $f(1) = 3 \times 2^{1-1} = 3$, and this is true because all three strings of length 1 each lack a double letter. Inductive Hypothesis: $f(n) = 3 \times 2^{n-1}$. Inductive Goal: $f(n+1) = 3 \times 2^n$. Let n be an arbitrary positive natural and consider the $3 \times 2^{n-1}$ strings in X with no double letter. Any string of length $n+1$ with no double letter must be made a string of length n with no double letter by appending another letter. For any such string w, there are exactly two choices how to append the letter – the two other than the last letter of w. So the number of strings of length $n+1$ in X is exactly twice $3 \times 2^{n-1} = 3 \times 2^n$ as desired.

- (b, 10) Prove that for any natural t such that $t \geq 8$, there exists at least one string of length t in Y. (**Hint:** Use strong induction, using three base cases.)

  We use strong induction on all naturals t with $t \geq 8$. First Base Case: $t = 8$, true by that string abcababc. Second Base Case: $t = 9$, true by the string abcabcabc. Third Base Case: $t = 10$, true by the string abcababcab. Strong Inductive Hypothesis: For any natural n, for any natural t with $8 \leq t \leq n$, there exists a string of length length t in Y, Inductive Step: If $8 \leq n \leq 10$, our goal is true by one of the base cases. Otherwise, if $n > 10$, then the IH tells us that there exists a string w of length $n - 2$ in Y. If we append abc to w, we form another string in Y of length $n + 1$.

- (c, 10 XC) Prove that Y is a subset of X. (**Hint:** Use induction on the star language Y. It may be useful to first prove by induction that no string in Y ends with an a.)
  Any string in Y is either λ or can be obtained by appending abcab or abc to some string already in Y. We first prove that no string in Y ends with a. For the base case, λ does not end with any letter and so does not end with a. For the inductive case, without even using the IH, we know that a string obtained by appending either string cannot end with a. Returning to the main proof, we prove the base case by noting that λ is in X since it has no double letter. Let the IH be that some arbitrary string w is in X. Our inductive step is to prove that both wabcab and wabc are in X. This is true because there no double letter in either string. The IH tells that there is no double letter in w. There is no double letter in either of the new strings. And there is no double letter by concatenating w and the new string, since w does not end with a and the new string starts with a.

Question 4 (40): Here is the usual question about Kleene's Theorem constructions. We begin with the language Y above, which is denoted by the regular expression $(abcab + abc)^*$.

- (a, 10) Using the construction from lecture, construct a λ-NFA N whose language is Y.

  We have ten states with start state 1, *only final state* 10, *and twelve edges:* $1\lambda2$, $2a3$, $2a7$, $2\lambda9$, $3b4$, $4c5$, $5a6$, $6b9$, $7a8$, $8c9$, $9\lambda2$, *and* $9\lambda10$. *This uses the star construction on the union construction on the ordinary NFA's for abcab and abc.*

- (b, 10) Using the construction from lecture, construct an ordinary NFA N' whose language is $L(N)$.

  We make the transitive closure of the λ-moves by adding $1\lambda9$, $1\lambda10$, *and* $2\lambda10$. *We make state* 1 *final because there is a λ-path from it to* 10. *Of the eight letter moves,* $2a3$ *causes itself and* $1a3$ *and* $9a3$, $2a7$ *causes itself and* $1a7$ *and* $9a7$, $3b4$, $4c5$, *and* $5a6$ *each cause itself,* $6b9$ *causes itself and* $6b2$ *and* $6b10$, $7b8$ *causes only itself, and* $8c9$ *causes itself,* $8c2$, *and* $8c10$. *This gives* 16 *letter moves.*

- (c, 10) Using the Subset Construction on N', construct a DFA whose language is $L(N')$. (**Hint:** You may simplify the diagram by saying where there are edges to a death state, without drawing them.)

  State $\{1\}$, *which is final, has an a-arrow to* $\{3,7\}$ *and no arrows to b or c. State* $\{3,7\}$, *which is non-final, has a b-arrow to* $\{4,8\}$ *and no arrows to a or c. State* $\emptyset$ *has all three arrows to itself. State* $\{4,8\}$, *which is non-final, has a c-arrow to* $\{2,5,9,10\}$, *and no arrows to a or b. State* $\{2,5,9,10\}$, *which is final, has an a-arrow to* $\{3,6,7\}$ *and no arrows to b or c. State* $\{3,6,7\}$, *which is non-final, has a b-arrow to* $\{2,4,8,9,10\}$, *and no arrows to a or c. State* $\{2,4,8,9,10\}$, *which is final, has an a-arrow to* $\{3,7\}$, *has a c-arrow to* $\{2,5,9,10\}$, *and no arrow to b. All these new edges go to previously seen nodes, so we are done with a seven-state DFA.*

- (d, 10) Using State Elimination, starting from D, construct a regular expression whose language is the same as $L(D)$. (Note that we already have a regular expression for Y, but we are asking you to use the construction. Your new regular expression may not be the same as the earlier one, but it should be equivalent, meaning that the two expressions will have the same language.)

  We begin by adding a new final state f, with λ-arrows to it from states 1, 25910, *and* 248910. *We don't need a new initial state since* 1 *has no arrows into it. We eliminate state* $\emptyset$, *adding no new arrows. We eliminate* 48 *to make* $(37, ab, 25910)$. *We eliminate* 367 *to make* $(25910, ab, 248910)$. *We eliminate* 37 *to make* $(1, abc, 25910)$ *and* $(248910, abc, 25910)$, *which merges with an existing arrow to produce* $(248910, c + abc, 25910)$. *We now eliminate* 248910 *to eliminate* $(25910, ab(c{+}abc), 25910$ *and* $(25910, ab, f)$. *The latter merges with an existing arrow to make* $(25910, \lambda + ab, f)$. *Finally we eliminate that last intermediate state* 25910 *to produce the regular expression* $\lambda + abc(ab(c + abc))^*(\lambda + ab)$

Question 5 (30): The following are fifteen true/false questions, with no explanation needed or wanted, no partial credit for wrong answers, and no penalty for guessing.

- (a) Because the divisibility relation over the natural numbers is reflexive, antisymmetric and transitive, it is therefore a total order.

  FALSE. It is only a partial order.

- (b) A bipartite graph of $2n$ nodes can contain as many as $\frac{(2n)^2}{4}$ edges.

  TRUE. A complete bipartite graph with n nodes on each has n^2 edges.

- (c) If the inverse of a statement is false, then the contrapositive could still be true.

  TRUE. It could either be true or false.

- (d) The following system of congruences can be simplified to a single congruence modulo $3 \times 8 \times 7 \times 15$, using the Chinese Remainder Theorem: $x \equiv 20 (\text{mod } 3)$, $x \equiv 18 (\text{mod } 8)$, $x \equiv 21 (\text{mod } 7)$, and $x \equiv 5 (\text{mod } 15)$.

  FALSE. The moduli are not pairwise relatively prime so the CRT does not apply. Either there will

- (e) Let $h_1(x)$ and $h_2(x)$ be two heuristics for the same search problem.
  Let $h_3(x) = \min(h_1(x), h_2(x))$. It is not possible for $h_3(x)$ to be admissible if both $h_1(x)$ and $h_2(x)$ are non-admissible heuristics.

  FALSE. Suppose that for each node one heuristic is too big and the other is 0, so that the minimum is zero, and thus is admissible.

- (f) Let L be the language denoted by the regular expression $(bbb + ccc)^* + (b + c)a^*$. Then the strings $bbbbb$ and $ccccbc$ are $L-$equivalent in the sense of the Myhill-Nerode Theorem.

  $\hat{\text{F}}$ALSE. They are distinguished by b..

- (g) There exists a two-way DFA which decides the unary language $\{1^{a^2} | a \geq 0\}$.

  FALSE. Two-way DFA's only decide regular languages, and this language is not regular.

- (h) There exists an NFA which decides the language $\{w \mid w$ contains an equal number of appearances of the substrings 01 and 10$\}$.

  TRUE. This is the set of strings that are either empty or have different letters in the first and last positions, so it is $\emptyset^* + 0\Sigma^*1 + 1\Sigma^*0$.

- (i) Let us define a Turing machine with a two-way infinite tape (2-way TM) to be identical to our standard TM, except the tape extends infinitely to the left as well as the right. Then these two-way TMs recognize the same class of languages as do standard TMs.

  TRUE. We could simulate a two-way TM with an ordinary TM by shifting the entire tape to the right every time we need to use a new cell to the left.

- (j) The following language EMPTY-DFA is not Turing decidable: It is the set of all DFA's D such that $L(D) = \emptyset$.

  FALSE. We could use DFS on TM to see whether is any path from the start state to any final state. If there is, D is not in this language — otherwise it is.

- (k) Suppose that we have a function f from pairs of strings to pairs of strings that is a reduction from the Halting Problem $H(M,w)$ to some arbitrary language $L(M,w)$, such that for any Turing machine M and any string w, we have that $(M,w) \in H \iff f(M,w) \in L$. From this we know that language L is undecidable.

  TRUE. If we could decide membership in L, we could use this reduction to decide H, which is impossible.

- (l) The class of regular languages is closed under reversal ($L^R = \{w^R | w \in L\}$).

  TRUE. We proved this in Section 5.5 of the textbook.

- (m) For any composite integer n, the sum of the prime factors of n is strictly less than n.

  TRUE. If n is the product of distinct primes, their sum is less than their product. If n involves powers of primes, then the sum is the same and the product is even bigger.

- (n) In any version of generic search, the search may terminate as soon as the goal vertex has been added to the open list.

  FALSE. UCS, for example, must continue until the goal vertex has been taken off the list.

- (o) If D is a minimal DFA with k states that decides language L, and the strings λ and a^{k+1} are L-equivalent, then the equivalence class of λ must contain at least three different strings.

  TRUE. The element $a^{2(k+1)}$ must also be equivalent to λ and a^{k+1}.