Roll No.: ____________________

CS 208 : Automata Theory and Logic Spring 2024

Quiz 1

Time: 21:00 - 23:30 *Max Marks:* 60

Instructions:

- ***Please write your roll number on all pages in the space provided at the top.***
- *Be brief, complete, and stick to what has been asked.*
- ***You must write your answer for every question only in the space allocated for answering the question. Answers written outside the allocated space risk not being graded.***
- ***You can use an extra answer book for rough calculations.***
- ***You must submit this question+answer book in its entirey along with any extra answer book for rough calculations (if you used one).***
- *Untidy presentation of answers, and random ramblings will be penalized by negative marks.*
- *Unless asked for explicitly, you may cite results/proofs covered in class without reproducing them.*
- ***If you need to make any assumptions, state them clearly.***
- ***Do not copy solutions from others. All detected cases of copying will be reported to DADAC with names and roll nos. of all involved. The stakes are high if you get reported to DADAC, so you are strongly advised not to risk this.***

DO NOT TURN THIS PAGE UNTIL YOU ARE ASKED TO.

THIS PAGE IS INTENTIONALLY LEFT BLANK

Roll No.: ____________________

1. **We love NNFs!** $\boxed{10}$

   Let P, Q, R be propositional variables.

   (a) Convert the formula $\big((P \rightarrow (Q \rightarrow R)) \rightarrow \neg(P \rightarrow (R \rightarrow Q))\big)$ to a semantically equivalent formula in Disjunctive Normal Form (DNF). Do not include cubes that contain both a literal and its negation in your DNF formula. $\boxed{5}$
   You must show all intermediate steps. Answers without steps will fetch no marks.

(b) Convert the formula $\big(\neg(P \lor (\neg Q \land R)) \to (\neg P \land (Q \lor \neg R))\big)$ to an equisatisfiable Conjunctive Normal Form (CNF) formula using Tseitin encoding. Do not include clauses that contain both a literal and its negation in your CNF formula. $\boxed{5}$
You must NOT simplify the given formula or check its satisfiability before applying Tseitin encoding. You must show all intermediate steps. Answers without steps or obtained after simplifying the given formula or after checking its satisfiability will fetch no marks. Answers that give an equisatisfiable formula without using Tseitin encoding will also fetch no marks.

2. **How expressive are you?** $\boxed{15}$

We know that propositional logic formulas are constructed using symbols in the set $\{\top, \bot, \neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ in addition to variables, parentheses and commas (if needed). It turns out that such a large set of symbols may not be needed. For example, the set $S' = \{\wedge, \neg\}$ suffices to construct a formula that is semantically equivalent to any propositional logic formula. Indeed, from DeMorgan's laws we know that $p_1 \vee p_2$ is semantically equivalent to $\neg(\neg p_1 \wedge \neg p_2)$ for every propositional variable (or sub-formula) p_1 and p_2.

Let S be a set of symbols that are used in addition to variables, parentheses and commas (if needed) to construct formulas. We say that S is ***propositionally expressive*** if for every propositional logic formula φ over variables $x_1, \ldots x_n$, there is a semantically equivalent formula over $x_1, \ldots x_n$ that uses only symbols from S in addition to variables, parentheses and commas (if needed). From what we have studied in class, it should be easy for you to see that $\{\wedge, \neg\}$ is propositionally expressive.

For purposes of this question, we define new ternary logic connectives α, β, γ and δ with the following semantics.

- $\alpha(p, q, r)$ evaluates to 1 (true) iff either both p and q are 1 (true) or p is 0 (false) and r is 1 (true). For example, $\alpha(1, 1, 0) = \alpha(0, 0, 1) = 1$ but $\alpha(1, 0, 1) = \alpha(0, 1, 0) = 0$. You can think of $\alpha(p, q, r)$ as intuitively implementing "if p then q else r".
- $\beta(p, q, r)$ evaluates to 1 (true) iff p, q and r all have the same truth value. Thus, $\beta(1, 1, 1) = \beta(0, 0, 0) = 1$, but $\beta(1, 0, 1) = \beta(1, 1, 0) = 0$. You can think of $\beta(p, q, r)$ as intuitively implementing "all of p, q, r are in consensus".
- $\gamma(p, q, r)$ evaluates to 1 (true) iff exactly one of p, q and r has the value 1 (true). Thus, $\gamma(1, 0, 0) = 1$ and $\gamma(1, 1, 0) = \gamma(0, 0, 0) = 0$.
- $\delta(p, q, r)$ evaluates to the same truth value as the majority of p, q and r. Thus, $\delta(1, 1, 0) = 1$ and $\delta(0, 1, 0) = 0$.

Given below are three sets S of symbols used to construct formulas. In each case you must indicate whether S is propositionally expressive, with justification.

You may use the fact that $\{\wedge, \neg\}$ is known to be propositionally expressive. Hence, for "Yes" answers, you need to show that $\neg p$ and $p \wedge q$ can be equivalently expressed using the symbols in S, for any propositional variables (or sub-formulas) p and q. For "No" answers, you must show that there is at least one formula that can be written using $\{\wedge, \neg\}$ but a semantically equivalent formula cannot be written using S.

(a) $S = \{\top, \bot, \alpha\}$ $\boxed{4}$

(b) $S = \{\top, \gamma\}$ $\boxed{4}$

(c) $S = \{\beta, \delta\}$ $\boxed{7}$

THIS PAGE IS INTENTIONALLY LEFT BLANK

Roll No.: ____________________

3. **Shipping with SAT** $\boxed{10}$

A shipping company has n cargo containers that must be transported via ships. Let $C = \{c_1, \ldots, c_n\}$ denote the containers. The company has m ships; let $S = \{s_1, \ldots s_m\}$ denote these ships. It turns out that not every ship can transport every container. Let $A_i \subseteq C$ be the set of containers that ***are allowed*** to be transported on ship i. Furthermore, each ship s_i has a ***maximum limit*** of l_i containers that it can transport. All l_i's are assumed to be non-negative integers.

The shipping company wants to find a set X of at most k $(0 < k \leq m)$ ships that can be used to transport all n containers, while loading each ship only with containers that are allowed on the ship, and without overloading each ship beyond the maximum number of containers it can transport.

As an example, consider $n = 4$, $m = 5$ and $l_1 = l_2 = l_4 = 1, l_3 = l_5 = 2$. Furthermore, suppose $A_1 = \{c_1, c_2\}$, $A_2 = \{c_2, c_3, c_4\}$, $A_3 = \{c_1, c_2, c_4\}$, $A_4 = \{c_2\}$ and $A_5 = \{c_2, c_4\}$. In this example, it is impossible to transport all 4 containers on only 2 ships. However, it is possible to transport all of them on 3 ships. For example, s_1 can be used to transport c_1, s_2 can be used to transport c_3 and s_5 can be used to transport c_2 and c_4. Hence $X = \{s_1, s_2, s_5\}$ is one possible solution the shipping company seeks.

We wish to use a satisfiability checker for NNF formulas to help the shipping company. Specifically, you must construct a propositional NNF formula φ such that

- Given n, m, k, $l_1, l_2, \ldots l_m$ where $0 \leq l_j \leq n$ for each $j \in \{1, \ldots m\}$, and the sets $A_1, A_2, \ldots A_m$, the formula φ can be constructed in time polynomial in m, n and k.
- There is a bijection between satisfying assignments of φ and distinct choices X of at most k ships that can transport all n containers, while respecting each ship's constraints. Note that this means φ must be unsatisfiable if it is impossible to transport all containers in at most k ships.

To construct the above formula, we will use propositional variables x_i for each $i \in \{1, \ldots m\}$, such that that x_i is true iff ship i is included in the set X of chosen ships.

You are free to use auxiliary propositional variables as you consider necessary. However, you must indicate the interpretation (what does the variable represent) for each such auxiliary variables. You are also free to use the cardinality constraints of the form $\sum_{p=u}^{v} b_p \leq w$ for propositional variables $b_u, \ldots b_v$ where $u \leq v$ and $0 \leq w \leq (v - u + 1)$. We have already discussed in Tutorial 1 how such a cardinality constraint can be encoded as a NNF formula in time polynomial in $(v - u)$ and w, possibly with the use of auxiliary propositional variables. Hence, if you are using such cardinality constraints, you don't need to explicitly write the NNF formula corresponding to $\sum_{p=u}^{v} b_p \leq w$, but can simply use $\left(\sum_{p=u}^{v} b_p \leq w\right)$ as a proxy for the NNF formula.

Roll No.: ______________________

4. **I think I saw you in Tuesday's lecture** $\boxed{10}$

   Draw a Deterministic Finite Automaton (DFA) for each of the following languages.

   (a) $\mathcal{L} := \{w \in \{a,b\}^* : 2 \text{ divides } n_a(w) \text{ and } 3 \text{ divides } n_b(w)\}$. Here $n_a(w)$ stands for the number of a's in w, and $n_b(w)$ stands for the number of b's in w. For example, $n_a(abbaab) = n_b(abbaab) = 3$. Therefore, $ababbaa \in \mathcal{L}$ but $aabababa \notin \mathcal{L}$. $\boxed{5}$

(b) $\mathcal{L} := \{w \in \{a,b,c\}^* :$ first and last letters of w are different$\}$. For example, $abbacb \in \mathcal{L}$ but $cbbabac \notin \mathcal{L}$. $\boxed{5}$

5. **To CNF or to DNF** $\boxed{15}$

Define the *length* of a CNF (or DNF) formula as the total number of all literals over all clauses (or all cubes, respectively) in the formula. For example, consider the CNF formula $\phi = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee x_3 \vee x_4)$. This formula has length $2+2+3 = 7$. Siimilarly, the length of the DNF formula $\psi = (x_1 \wedge x_2) \vee (\neg x_1 \wedge x_3 \wedge \neg x_4) \vee (\neg x_2 \wedge x_3 \wedge x_4)$ is $2 + 3 + 3 = 8$.

Show that there is a family of formulas $\mathcal{F} = \{\varphi_n \mid n \in \mathbb{N}\}$ such that

- Every φ_n is a DNF formula over $\mathcal{O}(n)$ propositional variables.
- Every φ_n has length in $\mathcal{O}(n)$.
- For every φ_n, there exists **no semantically equivalent** CNF formula ψ_n (over the same variables as φ_n) such that the length of ψ_n grows polynomially with n.

In order to answer this question, you must (a) clearly write the DNF formula φ_n, (b) show that its length is in $\mathcal{O}(n)$, and (c) prove that there exists **no semantically equivalent** CNF formula of length polynomial in n.

In order to score marks in this question, all three parts must be answered correctly.

Note: A formula has polynomial length if and only if *length* $\in \mathcal{O}(f(n))$ where f is some polynomial in n. You **MUST** prove why φ_n can't be equivalently represented by any polynomial length CNF formula.