CS 501: Formal Language Theory **Spring 2024**

Final Examination

Released: 5/15/2024, 6:00 pm EST Time Limit: 120 minutes Due: 5/15/2024, 8:00 pm EST

Note: LaTeX template courtesy of UC Berkeley EECS dept.

Instructions. This final contains seven questions on pages 1-9, for a total of 100 points with 5 extra credit. You have a total of 120 minutes to complete it. There will be a supplemental sheet with some definitions on it.

The final is an **individual effort**. You are required to write your entire attempt yourself, and are forbidden from consulting anyone else. Failure to abide by this will result in immediate failure from the course, among other consequences.

- This is a closed-book exam, with no books, notes, calculators, or collaboration.

Submissions. Please write your answers on the test sheet. You may use the backs of pages, but let us know in the indicated place for each question where we can find the rest of your answer. Page 10 and 11 is a supplemental sheet with useful information – do not put answers on it.

---

1. **(10 × 2 points) Unjustified True/False Questions.** For each of the following questions, indicate simply whether it is TRUE or FALSE. *No justification needed or wanted.*

   (a) The circuit classes in the NC hierarchy satisfy the inclusions $NC^0 \subseteq AC^0 \subseteq NC^1 \subseteq AC^1 \subseteq \ldots$, and none of these inclusions are known to be strict.

   (b) If a language Y and its complement $\overline{Y}$ are both context-free, then it is regular.

   (c) If there exists a linear-time algorithm for the vertex cover problem VC, then the classes NP and co-NP must be equal.

   (d) It is not known whether the TQBF problem (the set of true quantified boolean formulas) is solvable in polynomial time.

   (e) If a language is defined by a context-sensitive grammar, it can be decided by a deterministic Turing machine using $O(n^2)$ space.

   (f) If $|\Sigma| = 1$, the Post Correspondence Problem over the alphabet Σ is Turing decidable.

   (g) With $\Sigma = \{0, 1\}$, any sufficiently long palindrome fails to be incompressible.

   (h) The Regular Language Pumping Lemma may be used to prove a language to be regular, by showing that every string in the language satisfies the conclusion of the lemma.

   (i) Assuming the result of Question 4 below, the class POLYNBP is closed under complementation.

   (j) The set SELF of all Turing machines M that output their own descriptions is a TR language, but is not TD.

2. **(5 $\times$ 6 points) Justified True/False Questions.** For each of the following questions, indicate whether it is TRUE or FALSE, *and provide a brief justification (i.e. either a proof or a counterexample).*

   (a) On the midterm, we defined a **computable collection of TR languages** $(L_1, L_2, L_3, \ldots)$, where each language L_i is the language of some TM M_i, and there is a single computable function q that produces a description of M_i on input i. Let INT be the set of all languages that are intersections of computable collections of TR languages. Then INT is complete for the Π_2 level of the Arithmetic Hierarchy.

   (b) We have seen that the language REACH, of triples (G, s, t) such that G is a directed graph and there is a path from s to t in G, is NL-complete under $\leq_L$ reductions. Define the language DAG-REACH to be the subset of REACH where each graph G must be a directed acyclic graph, with its nodes numbered such that any edge goes from smaller to larger node numbers. Then DAG-REACH is also NL-complete under $\leq_L$ reductions.

(c) The language $OVERLAP_{CFG}$ is the set of all pairs (G, H) of CFG's such that $L(G) \cap L(H) \neq \emptyset$. Then the language $OVERLAP_{CFG}$ is Turing decidable.

(d) Define a **Sillier TM** to be a one-tape Turing machine M such that if q is a state and a and b are any two letters in M's tape alphabet, then $\delta(q, a) = \delta(q, b)$. Then the language $L(M)$ of any Sillier TM M is finite, and thus the language $A_{SillierTM}$ is TD.

(e) Let S be an arbitrary *finite* set of undirected graphs. Define the language $Contains_S$ to be the set of undirected graphs G such that there exists a graph H in S such that H is a subgraph of G (see the definition on the supplemental sheet). Then, assuming $P \neq NP$, the language $Contains_S$ is not in the class P.

3. **(10 points + 5 XC) Wordle and Regular Languages.** Let W be the set of all five-letter strings over the alphabet $\Sigma = \{a, b, \ldots, z\}$ and let D be a arbitrary subset of W. We define the language C_D to be the set of all strings (of whatever length) that contain five consecutive letters that form a string in D.

   - (a, 5) Prove, by any method, that for any D, the language C_D is regular.
   - (b, 5) Prove a finite upper bound on the number of Myhill-Nerode classes in C_D, for any D. (Note that this result implies that of part (a), but you may want to do (a) separately in case your solution to (b) is wrong.)
   - (c, 5XC) Prove that for any D with $|D| = n$, there are at most $5n + 1$ Myhill-Nerode classes for C_D. (Again, solving this also solves both (a) and (b), but there are easier ways to solve (b).)

4. **(10 points) Nondeterministic Branching Programs.** In Discussion #10, we defined **branching programs** and the language POLYBP of languages defined by log-space uniform polynomial-size branching programs. Here we define **nondeterministic branching programs (NBP's)**, which are like branching programs but where a non-leaf node may have any number of 0-edges and any number of 1 edges, as well as **free edges** that may be taken whatever the input variable is. An input is in the language of the NBP if it is possible to go from the start node to an accepting leaf, using either ordinary edges matching the input or free edges. We define the class POLYNBP to the the set of languages of log-space uniform NBP families whose size (number of nodes) is bounded by a polynomial in their number of inputs. Prove that the class POLYNBP is equal to the class NL.

5. **(10 points) Classroom Scheduling is NP-Complete.** A University registrar has n students and m courses for which she needs to assign final exam times, using k time slots. If x and y are two different courses, and there is any student in both courses, she may not give them the same time slot. Let SCHED be the set of tuples (S, C, A, k) such that:

   - S is a list of students,
   - C is a list of courses,
   - A is the set of pairs (s, c) in $S \times C$ such student s is in course c, and
   - k is the number of time slots, such that
   - It is possible to assign each course in C to a time slot with no student being assigned two courses with the same time slot.

   Prove that this language is NP-complete.

6. **(10 points) How Many a's and b's?** Let $\Sigma = \{a, b\}$ and define a language X that is a subset of the regular language a^*b^*. A string a^ib^j is in X if and only if $j \leq i \leq 2j$. That is, there are at least as many a's as there are b's, and there are not more a's than twice as many as there are b's. Find **both**:

   - A context-free grammar for X, and
   - A pushdown automaton for X.

   You may use standard constructions to convert one to the other, but we want *explicit* examples for both the CFG and PDA. Producing one and just claiming that the other exists will get you no points for the other.

7. **(10 points) Implicitly Defined Circuits.** We're going to define a family of Boolean circuits, but they will be too large to be given as input in the normal way. An **Implicitly Defined Circuit (IDC)** with n inputs is defined by a poly-time Turing machine M which can be used to get an exponential number of NC^0 circuits, each with n inputs and n outputs. (Recall that an NC^0 circuit is one where each output depends on only a constant number of inputs.) For any input size n and any number i with $1 \leq i \leq 2^{p(n)}$, M outputs a description of the NC^0 circuit C_i. (Here $p(n)$ is a fixed polynomial in n.) The IDC is the composition of all the circuits, so that the n inputs feed into C_1, which feed into C_2, and so on, until the rightmost output bit of $C_{2^{p(n)}}$ is the output of the IDC.

   The language IDCVAL is the set of pairs (M, w) where M is such a Turing machine, w is a binary string, and the corresponding IDC accept w. Prove that the language IDCVAL is PSPACE-complete.

Supplemental Sheet for COMPSCI 501 Final Exam, Spring 2024

Language A_{TM} (similarly A_{REG}, etc.) Set of pairs (M, w) such that M accepts w

Language ALL_{TM}**:** (similarly ALL_{REG}, etc.) Set of machines that accept all possible strings over their alphabet

Arithmetic Hierarchy: Classes of languages defined by first-order formulas where the quantifier-free part is TD. Σ_i means i quantifiers starting with $\exists$, Π_i means i quantifiers starting with $\forall$.

Branching Program: A DAG where each node has out-degree zero or two, leaves are labeled “accept” or ”reject”, the other nodes are each labeled with one of n input variables, and the edges out of that node are labeled 0 or 1. This defines a path from a start node to some leaf, depending on the input values.

Circuit Complexity: $PSIZE$ is all languages X for which there is a uniform family of circuits, where circuit C_n decides membership in X for strings of length n, and C_n has $n^{O(1)}$ nodes. If the circuit depth is also bounded by $O(\log^i n)$, the language is in the class NC^i (if the circuits have fan-in two) or AC^i (if the circuits have unbounded fan-in).

Computable Function: Function f from strings to strings such that some Turing machine, on any input w, always halts with $f(w)$ on its tape

Context-Free Grammar (CFG): Grammar where rules allow single non-terminals to be replaced by strings

Context-Free Language (CFL): Definable by a context-free grammar or a PDA

Context-Sensitive Grammar: Grammar where rules are of the form $S \to \varepsilon$ or $\alpha A \beta \to \alpha\beta\gamma$, where A is a non-terminal, α, β, and γ are strings of terminals or non-terminals, and $\gamma \neq \varepsilon$

co-TR: A language A is co-TR if and only if its complement $\overline{A}$ is TR (similarly co-NP, etc.)

Language E_{TM}**:** (similarly E_{REG}, etc.) Set of machines with empty languages

Language EQ_{TM}**:** (similarly EQ_{REG}, etc.) Set of pairs of machines whose languages are equal

Incompressible: A binary string is incompressible if is has no description (in terms of a TM and input string for it) that is shorter than itself.

Log-Space Reduction ($\leq_L$**):** Like mapping reduction except that the function f is log-space computable

Mapping Reduction ($\leq_m$**):** $A \leq_m B$ means that there exists a computable function f such that $\forall w : w \in A \leftrightarrow f(w) \in B$

NP-Complete Languages: You may assume without proof that all these are NP-complete:

- SAT, the set of satisfiable boolean formulas
- 3-SAT, the set of 3-CNF boolean formulas such that no two adjacent vertices have the same color
- $CIRCUIT$-SAT, the set of satisfiable boolean circuits
- $CLIQUE$, the set of pairs (G, k) with a k-clique in the undirected graph G
- VC, the set of pairs (G, k) where there exists a k-vertex cover in the undirected graph G
- 3-COLOR, the set of undirected graphs whose vertices can be colored with three colors
- $HAMPATH$, the set of directed graphs G with a path from one node to another using every node once
- $UHAMPATH$, the set of undirected graphs G with such a path
- $SUBSET$-SUM, the set of sequences $(s_1, \ldots, s_k, t)$ where there is a subset of the s_i's summing to t

Poly-Time Reduction ($\leq_p$): Like mapping reduction except the function f is poly-time computable

Pushdown Automaton (PDA): Nondeterministic finite-state machine with an added stack

REACH: The language $\langle G, s, t\rangle$ such that G is a directed graph in which there is a path from node s to node t

Regular Language: Can be defined by a DFA, NFA, or regular expression

Space Complexity: $DSPACE(f)$ is the set of languages decided by a TM using $O(f)$ steps, $NSPACE$ similarly for NDTM's, $PSPACE = \cup_k DSPACE(n^k)$, $NPSPACE = \cup_k NSPACE(n^k)$, $L = DSPACE(\log n)$, $NL = NSPACE(\log n)$

Subgraph: An undirected graph H is a subgraph of G if the nodes of H are a subset of the nodes of G, and each edges of H is also an edge of G.

Time Complexity: $DTIME(f)$ is the set of languages decided by a TM using $O(f)$ steps, $NTIME$ similarly for NDTM's, $P = \cup_k DTIME(n^k)$, $NP = \cup_k NTIME(n^k)$

Turing Decidable (TD): Is the language of a TM that always halts

Turing Recognizable (TR): Is the language of any TM