NAME:_____ SPIRE ID:_____

CS 501: Formal Language Theory **Spring 2026**

Final Examination

Released: 5/13/2026, 6:00 pm EST Time Limit: 120 minutes Due: 5/13/2026, 8:00 pm EST

Note: LaTeX template courtesy of UC Berkeley EECS dept.

Instructions. This final contains eight questions on pages 1-10, 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 definitions.

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. Pages 11 and 12 are a supplemental sheet with useful information – do not put answers on it.

Supplementary Definitions: Several questions on the exam use two definitions which we will give here and also include on the supplemental sheet.

A **groupoid** is a set S equipped with a binary operation, that is, a function from $S \times S$ to S. We will call the operation "multiplication", but we make *no other assumptions* about this operation—for example, if we have a sequence of elements to multiply together, we must consider the possibility that there might be multiple answers because of the order in which the terms are multiplied.

Let G be a finite non-empty groupoid. Assume that the symbols "(" and ")" are not in G. A **fully parenthesized term** over a groupoid G is a string over the alphabet $\Sigma = G \cup \{(,)\}$ defined recursively:

- Any element $\sigma \in G$ is an FP term.
- If u and v are FP terms, then "(uv)" is an FP term.

So, for example, if G has elements a and b, then a, $(a(ab))$, and $((aa)(ba))$ are FP terms, but ϵ, ab, and $a(ab)$ are not. Any FP term can be evaluated unambiguously as an element of G, but different FP terms with the same letters may evaluate to different results.

The language $GGEN$ is the set of triples (G, s, t) where G is a groupoid on some set S, given as a multiplication table, and s and t are elements of S, such that there exists a positive integer n and a way to multiply n copies of s together for a result of t.

A **monoid** is a groupoid that also satisfies two additional assumptions—the operation is **associative** (so that $(xy)z = x(yz)$ for any elements x, y, and z, and there is an **identity element** e such that $ex = xe = x$ for any element x.

An **LBA-function** is a linear bounded automaton with input alphabet $\{0, 1\}$ and a tape alphabet including $\{0, 1, \#\}$ and possible other letters. On input $u\#v$, for binary strings u and v, it halts with exactly two # symbols and only 0's and 1's between them. We interpret this as multiplying u and v to get the string w between the #'s. This defines a groupoid on the set $\{0, 1\}^*$.

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) Let Q be the set of all strings in $\{a, b\}^*$ that contain a string of consecutive a's longer than $|w|/2$. Then Q is a context-free language.

   (b) Because DFAs and NFAs have the same computational power, the languages A_{DFA} and A_{NFA} have the same time and space complexity.

   (c) The set I of incompressible strings is in the Π_1 class of the Arithmetic Hierarchy.

   (d) Let X be any language and let Y be a regular language. If $X \cap Y$ is a CFL, then X must also be a CFL.

   (e) Let N be a nondeterministic Turing machine with the property that given integer n, there exists a number $t(n)$ such that no matter what choices it makes, it cannot continue running for more than $t(n)$ steps on any input of length n. Then $L(N)$ is a TD language.

   (f) Let M be any Turing machine such that $L(M) \in \mathbf{P}$. Then M never repeats a configuration on any input w.

   (g) There exists a Turing machine that accepts any string w if and only if w is not its own description.

   (h) Assuming that $\mathbf{P} \neq \mathbf{NP}$, the language DNF-SAT is not in $\mathbf{P}$.

   (i) Using the Alternation Theorem, we can prove that if $\mathbf{P} \neq \mathbf{NP}$, then $\mathbf{PSPACE} \neq \mathbf{NPSPACE}$.

   (j) A groupoid S is called **idempotent** if for any element $\sigma \in G$, $\sigma\sigma = \sigma$. Let $IGEN$ be the version of the $GGEN$ problem where the groupoid must be idempotent. Then if $\mathbf{L} \neq \mathbf{P}$, then the language $IGEN$ is not in $\mathbf{L}$.

2. **(5 × 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) Let $\Sigma = \{a, b, \ldots, z\}$ (with exactly 26 letters) and let MODCOUNT be the set of strings w over Σ such that there do not exist two different letters α and β, such that the number of occurrences of α in w and the number of occurrences of β in w are congruent modulo 26. Then MODCOUNT is not a regular language.

   (b) Recall the definition of "FP terms" for a groupoid G. For any groupoid, define FPT_G to be the set of FP terms over G. Then FPT_G is a context-free language.

   (c) Let X be any TR language and let Y be any TD language. Then $X \cap Y$ is TD if and only if X is TD.

(d) There exists a log-space transducer that takes any boolean circuit, with n boolean inputs, and returns an equivalent monotone circuit (meaning that it computes the same function of its input variables).

(e) Assume that $\mathbf{P} \neq \mathbf{NP}$. Then the language A_{LBA} could possibly be in the class $\mathbf{P}$.

3. **(10 points) An Infinite Groupoid**

   Consider the groupoid on the infinite set $\{0,1\}^*$ defined by a given LBA-function. The language LBA-GEN is the set of triples (M,u,v) such that M defines an LBA-function and there exists a positive integer n such that some product of n copies of u, with some bracketing, results in v.

   Prove that the language LBA-GEN is Turing recognizable but not Turing decidable.

4. **(10 points) Covering a Clique**

   If $G = (V, E)$ is an undirected graph, a **clique cover of size** k of G is a set of k subsets of V, $\{S_1, S_2, \ldots, S_k\}$, such that each S_i is a clique in G and $S_1 \cup \ldots \cup S_k = V$. The language CLIQUE-COVER is the set of pairs (G, k) such that G is an undirected graph and G has a clique cover of size k.

   Prove that CLIQUE-COVER is **NP**-complete. There are a number of ways to show that this language is **NP**-hard, and you may use any of them, but here is a suggestion. On HW#5, you proved that the problem k-COLOR, the set of pairs (G, k) such that G is an undirected graph that can be colored with k colors, is **NP**-complete. (So you may assume here that k-COLOR is **NP**-complete.) Can you find a relationship between CLIQUE-COVER and k-COLOR?

5. **(10 points)** $GGEN$ **is complete for P**

   The language $GGEN$ was defined above to be the set of triples (G, s, t) such that G is a (finite) groupoid over a set S, given by a multiplication table, s and t are elements of S, and there is positive integer n such that a product of n copies of s, with some bracketing, results in t.

   Prove that $GGEN$ is complete for the class **P** under $\leq_L$ reductions. (**Hint:** Reduce from $MCVP$, the monotone version of CVP, which we have proved to be **P**-complete.)

6. **(10 points) Monoid Generation in Logspace**

We defined a **monoid** above as a groupoid that is associative and contains an identity element. $MGEN$ is the special case of $GGEN$ where the given groupoid G is also a monoid, so that (M, s, t) is the set of triples such that M is a monoid, s and t are elements of M, and there exists a postive integer n such that the product of n copies of s results in t.

(a, 5) Prove that the language MON_{TABLE}, the set of $n \times n$ multiplication tables that represent monoids, can be decided in the class $\mathbf{AC}^0$. (Your circuit family will likely be log-space uniform, but we don't ask you to prove this here.) (Note also that only certain input sizes can occur in this problem, as a table for an m-element groupoid has m^2 table entries, each of about $\log m$.)

(b, 5) Prove that $MGEN$, as defined above, is in the class $\mathbf{L}$. (It would be complete for $\mathbf{L}$ if we had an appropriate reducibility available within $\mathbf{L}$.)

7. **(10 points) Regular Languages in ALOGTIME**

   Recall that the class **ALOGTIME** is the set of languages of alternating Turing machines that run in $O(\log n)$ time. These machines have a **read-only input tape** of length n and access it through an **index access tape** of length $\log n$, initially set to all 0's. At every step, the machine can see the input bit in the position currently written on the input access tape.

   Prove that the set of regular languages over $\{0, 1\}$ is a subset of **ALOGTIME**.

   (**Hint:** Note that the ATM is able to see only a small fraction of the input bits, but the players in the ATM game know all of the bits. The ATM only needs to access only one bit of the input at the end of the game, where the address of the bit to check is determined by the interaction between the players.)

8. **(5 points extra credit) FP Term Validity in Logspace**

   Let G be a fixed finite groupoid. Prove that the language FPT_G, defined in Question 2b, is in the class **L**.

Supplemental Sheet for COMPSCI 501 Final Exam, Spring 2025

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

Language A_P: (similarly A_{NP}, A_{PSPACE}, etc.) Set of triples $(M, w, 1^t)$ such that M accepts w in at most t steps

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 $\mathbf{NC}^i$ (if the circuits have fan-in two) or $\mathbf{AC}^i$ (if the circuits have unbounded fan-in).

Circuit Value Problem (CVP): The set of pairs (C, x) where C is a boolean circuit with n boolean inputs, x is a binary string of length n, and C on x evaluates to true

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.)

FP term: A language defined over a groupoid G, where the input alphabet consists of G's elements, "(", and ")", and the strings are defined as on Page 1

FPT_G: The language of valid FP terms for a given groupoid G

$GGEN$: The set of triples (G, s, t) where G is a groupoid, given as a multiplication table, s and t are elements, and there is some way to multiply some number of copies of s, with some bracketing, to obtain t

Groupoid: A set S with a binary operation on S, not necessarily associative

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.

Linear-Bounded Automaton (LBA): A one-tape TM that can never move its head off of the area of the tape originally containing its input

LBA-function: A binary operation on the set of all binary strings, defined by an LBA as defined on page 1, forming a groupoid

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

Log-Space Transducer: The function in a log-space reduction, with a red-only input tape and a write-only output tape along with its work tape of size $O(\log n)$

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

Monotone Boolean Function: A function from n inputs to one output such that changing a value of any input can only increase the output value

Monoid: A groupoid whose operation is associative and has an identity element

Monotone Circuit: A boolean circuit that uses only AND and OR gates, never using NOT gates

Monotone Circuit Value Problem (MCVP): The special case of CVP where C is a monotone circuit.

NP-Complete Languages: You may assume without proof that all these are NP-complete (except for CLIQUE COVER which is the subject of Question 4):

- SAT, the set of satisfiable boolean formulas
- 3-SAT, the set of 3-CNF boolean formulas
- CIRCUIT-SAT, the set of satisfiable boolean circuits
- CLIQUE, the set of pairs (G, k) with a k-clique in the undirected graph G
- CLIQUE COVER, the set of pairs (G, k) such that the vertices of G are the union of k cliques in 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
- k-color, the set of pairs (G, k) such that G is an undirected graph that can be colored with at most k colors
- HAMPATH, the set of directed graphs G with a path from one node to another using every node once
- 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 $O(f)$ tape cells, $NSPACE$ similarly for NDTM's, $\mathbf{PSPACE} = \cup_k DSPACE(n^k)$, $\mathbf{NPSPACE} = \cup_k NSPACE(n^k)$, $\mathbf{L} = DSPACE(\log n)$, $\mathbf{NL} = NSPACE(\log n)$

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 edge 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, $\mathbf{P} = \cup_k DTIME(n^k)$, $\mathbf{NP} = \cup_k NTIME(n^k)$

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

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