Gradiance Online Accelerated Learning

Zayd

- Home Page
- Assignments Due
- Progress Report
- Handouts
- Tutorials
- Homeworks
- Lab Projects
- Log Out

Help

Submission number: 89164
Submission certificate: HB273033
Submission time: 2014-05-19 00:04:37 PST (GMT - 8:00)

Number of questions: 25
Positive points per question: 4.0
Negative points per question: 0.0
Your score: 100

1. Design the minimum-state DFA that accepts all and only the strings of 0's and 1's that end in 010. To verify that you have designed the correct automaton, we will ask you to identify the true statement in a list of choices. These choices will involve:

1. The number of *loops* (transitions from a state to itself).
2. The number of transitions into a state (including loops) on input 1.
3. The number of transitions into a state (including loops) on input 0.

Count the number of transitions into each of your states ("in-transitions") on input 1 and also on input 0. Count the number of loops on input 1 and on input 0. Then, find the true statement in the following list.

a) There is one state that has one in-transition on input 1.

b) There are two loops on input 1 and no loop on input 0.

c) There is one loop on input 0 and no loop on input 1.

d) There are two states that have two in-transitions on input 1.

Answer submitted: **d)**

You have answered the question correctly.

2. The homomorphism *h* is defined by h(a) = 01 and h(b) = 10. What is h (babb)?

a) 10101010

b) babb

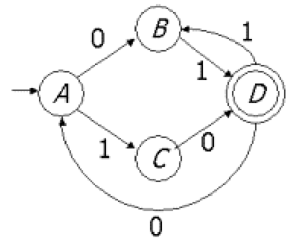
c) 100110

d) 10011010

Answer submitted: **d)**

You have answered the question correctly.

3. Here is a finite automaton:



Which of the following regular expressions defines the same language as the finite automaton? Hint: each of the correct choices uses component expressions. Some of these components are:

1. The ways to get from A to D without going through D.
2. The ways to get from D to itself, without going through D.
3. The ways to get from A to itself, without going through A.

It helps to write down these expressions first, and then look for an expression that defines all the paths from A to D.

a) ((01+10)(11)*0)*(01+10)(11)*

b) ((01+10)(11)*0)*(01+10)

c) ((01+10)0+11)*(01+10)(11)*

d) (01+10)(0(01+10))*(11)*

Answer submitted: **a)**

You have answered the question correctly.

4. The Turing machine M has:

- States q and p; q is the start state.
- Tape symbols 0, 1, and B; 0 and 1 are input symbols, and B is the blank.
- The following next-move function:

State	Tape Symbol	Move
q	0	(q,0,R)
q	1	(p,0,R)
q	B	(q,B,R)
p	0	(q,0,L)
p	1	none (halt)
p	B	(q,0,L)

Your problem is to describe the property of an input string that makes M halt. Identify a string that makes M halt from the list below.

a) 001010

b) 10101

c) 0110

d) 0100

Answer submitted: **c)**

You have answered the question correctly.

5. Let G be the grammar:

```
S → SS | (S) | ε
```

L(G) is the language BP of all strings of balanced parentheses, that is, those strings that could appear in a well-formed arithmetic expression. We want to prove that L(G) = BP, which requires two inductive proofs:

1. If w is in L(G), then w is in BP.
2. If w is in BP, then w is in L(G).

We shall here prove only the first. You will see below a sequence of steps in the proof, each with a reason left out. These reasons belong to one of three classes:

A)
Use of the inductive hypothesis.

B)
Reasoning about properties of grammars, e.g., that every derivation has at least one step.

C)
Reasoning about properties of strings, e.g., that every string is longer than any of its proper substrings.

The proof is an induction on the number of steps in the derivation of w. You should decide on the reason for each step in the proof below, and then identify from the available choices a correct pair consisting of a step and a kind of reason (A, B, or C).

Basis: One step.

(1)
The only 1-step derivation of a terminal string is S => ε because ________

(2)
ε is in BP because ________
Induction: An n-step derivation for some n>1.

(3)
The derivation $S =>^{n}$ w is either of the form
(a) $S => SS =>^{n-1}$ w or of the form
(b) $S => (S) =>^{n-1}$ w
because ________
Case (a):

(4)
w = xy, for some strings x and y such that $S =>^{p}$ x and $S =>^{q}$ y, where p<n and q<n because ________

(5)
x is in BP because ________

(6)
y is in BP because ________

(7)
w is in BP because ________
Case (b):

(8)
w = (z) for some string z such that $S =>^{n-1}$ z because ________

(9)
z is in BP because ________

(10)
w is in BP because ________

a) (6) for reason A
b) (5) for reason B
c) (2) for reason B
d) (8) for reason A

Answer submitted: **a)**

You have answered the question correctly.

6. For the grammar:

```
S → AB | CD
A → BC | a
B → AC | C
C → AB | CD
D → AC | d
```

1. Find the generating symbols. Recall, a grammar symbol is *generating* if there is a deriviation of at least one terminal string, starting with that symbol.
2. Eliminate all *useless productions* --- those that contain at least one symbol that is not a generating symbol.
3. In the resulting grammar, eliminate all symbols that are not *reachable* --- they appear in no string derived from S.

In the list below, you will find several statements about which symbols are generating, which are reachable, and which productions are useless. Select the one that is FALSE.

a) B → C is useless.

b) S is not generating.

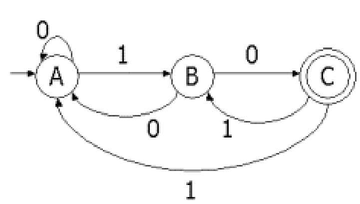
c) C → CD is useless.

d) *a* is not generating.

Answer submitted: **d)**

You have answered the question correctly.

7. Convert the following nondeterministic finite automaton:



to a DFA, including the dead state, if necessary. Which of the following sets of NFA states is **not** a state of the DFA that is accessible from the start state of the DFA?

a) {B}

b) {A,C}

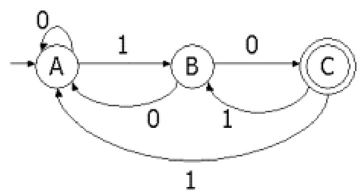
c) {}

d) {C}

Answer submitted: **d)**

You have answered the question correctly.

8. When we convert an automaton to a regular expression, we need to build expressions for the labels along paths from one state to another state that do not go through certain other states. Below is a nondeterministic finite automaton with three states. For each of the six orders of the three states, find regular expressions that give the set of labels along all paths from the first state to the second state that never go through the third state.



Then identify one of these expressions from the list of choices below.

a) (0+10)*1 represents the paths from A to B that do not go through C.

b) ((1+11)0)*1 represents the paths from C to B that do not go through A.

c) 1 represents the paths from C to A that do not go through B.

d) 1(0+1010)* represents the paths from C to A that do not go through B.

Answer submitted: **a)**

You have answered the question correctly.

9. Consider the grammar G and the language L:

G: S → AB | a | abC, A → b, C → abC | c

L: {w | w a string of a's, b's, and c's with an equal number of a's and b's}.

Grammar G does not define language L. To prove, we use a string that either is produced by G and not contained in L or is contained in L but is not produced by G. Which string can be used to prove it?

a) cacacab

b) ccababab

c) cacabbb

d) abac

Answer submitted: **b)**

You have answered the question correctly.

10. The language L = {ss | s is a string of a's and b's} is not a context-free language. In order to prove that L is not context-free we need to show that for every integer n, there is some string z in L, of length at least n, such that no matter how we break z up as $z = uvwxy$, subject to the constraints |vwx| ≤ n and |vx| > 0, there is some $i \geq 0$ such that uv^iwx^iy is not in L.

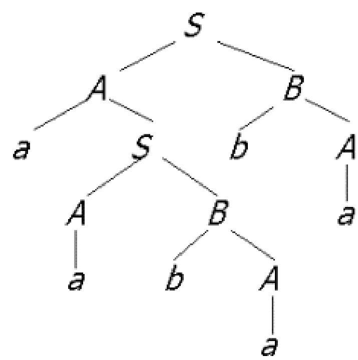
Let us focus on a particular z = aabaaaba and n = 7. It turns out that this is the wrong choice of z for n = 7, since there are some ways to break z up for which we can find the desired *i*, and for others, we cannot. Identify from the list below the choice of u,v,w,x,y for which there is an *i* that makes uv^iwx^iy not be in L. We show the breakup of aabaaaba by placing four |'s among the a's and b's. The resulting five pieces (some of which may be empty), are the five strings. For instance, aa|b||aaaba| means u=aa, v=b, w=ε, x=aaaba, and y=ε.

a) |a|abaa|a|ba
b) |a|ab|a|aaba
c) aab|a|a|a|ba
d) a|a|ba|a|aba

Answer submitted: **c)**

You have answered the question correctly.

11. The parse tree below represents a leftmost derivation according to the grammar `S → AB, A → aS|a, B → bA`.

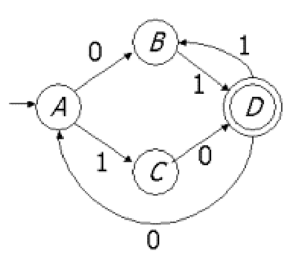


Which of the following is a left-sentential form in this derivation?
a) aaABBB
b) aaBB
c) aAbaba
d) aabAba

Answer submitted: **b)**

You have answered the question correctly.

12. Examine the following DFA:



Identify in the list below the string that this automaton accepts.
a) 0111010

b) 010011

c) 1011011

d) 01011

Answer submitted: **a)**

You have answered the question correctly.

13. Consider the pushdown automaton with the following transition rules:

1. $\delta(q,0,Z_0) = \{(q,XZ_0)\}$
2. $\delta(q,0,X) = \{(q,XX)\}$
3. $\delta(q,1,X) = \{(q,X)\}$
4. $\delta(q,\varepsilon,X) = \{(p,\varepsilon)\}$
5. $\delta(p,\varepsilon,X) = \{(p,\varepsilon)\}$
6. $\delta(p,1,X) = \{(p,XX)\}$
7. $\delta(p,1,Z_0) = \{(p,\varepsilon)\}$

The start state is *q*. For which of the following inputs can the PDA first enter state *p* with the input empty and the stack containing XXZ_0 [i.e., the ID (p,ε,XXZ_0)]?

a) 010101

b) 1001101

c) 0111011

d) 0100110

Answer submitted: **a)**

You have answered the question correctly.

14. The binary string 0101111 is a member of which of the following problems? Remember, a "problem" is a language whose strings represent the cases of a problem that have the answer "yes." In this question, you should assume that all languages are sets of binary strings interpreted as base-2 integers. The exception is the problem of finding *palindromes*, which are strings that are identical when reversed, like 0110110, regardless of their numerical value.

a) Is the given string greater than 50?

b) Is the given string a palindrome?

c) Is the given string a perfect cube?

d) Is the given string not a perfect square?

Answer submitted: **d)**

You have answered the question correctly.

15. Programming languages are often described using an extended form of context-free grammar, where curly brackets are used to denote a construct that can repeat 0, 1, 2, or any number of times. For example, $A \rightarrow B\{C\}D$ says that an *A* can be replaced by a *B* and a *D*, with any number of *C*'s (including 0) between them. This notation does not allow us to describe anything but context-free languages, since an extended production can always be replaced by several conventional productions.

Suppose a grammar has the extended production:

```
A → B0{1C}D1
```

Convert this extended production to conventional productions. Identify, from the list below, the conventional productions that are equivalent to the extended production above.

a) $A \rightarrow B01CA_1D1$
$A_1 \rightarrow 1CA_1 \mid \varepsilon$

b) $A \rightarrow B0A_1D1$
$A_1 \rightarrow 1CA_1 \mid \varepsilon$

c) $A \rightarrow B0D1 \mid B01CD1 \mid B01C1CD1 \mid B01C1C1CD1 \mid \ldots$

d) $A \rightarrow B0A_1D1$
$A_1 \rightarrow A_11C \mid 1C$

Answer submitted: **b)**

You have answered the question correctly.

16. A *linear grammar* is a context-free grammar in which no production body has more than one occurrence of one variable. For example, $A \rightarrow 0B1$ or $A \rightarrow 001$ could be productions of a linear grammar, but $A \rightarrow BB$ or $A \rightarrow A0B$ could not. A *linear language* is a language that has at least one linear grammar.

The following statement is false: "The concatenation of two linear languages is a linear language." To prove it we use a counterexample: We give two linear languages L_1 and L_2 and show that their concatenation is not a linear language. Which of the following can serve as a counterexample?

a) $L_1 =\{w|w=(ab)^na^nb^n$, where n is a positive integer$\}$
$L_2=\{w|w=c(aaa)^n(ab)^n$, where n is a positive integer$\}$

b) $L_1 =\{w|w=aaac^{n-1}(ab)$, where n is a positive integer$\}$
$L_2=\{w|w=c(aaa)^n$, where n is a positive integer$\}$

c) $L_1 =\{w|w=a^nb^n$, where n is a positive integer$\}$
$L_2=\{w|w=(abb)^n(ab)^n$, where n is a positive integer$\}$

d) $L_1 =\{w|w=aaa(aba)^n(ab)$, where n is a positive integer$\}$
$L_2=\{w|w=c(aaa)^n$, where n is a positive integer$\}$

Answer submitted: **c)**

You have answered the question correctly.

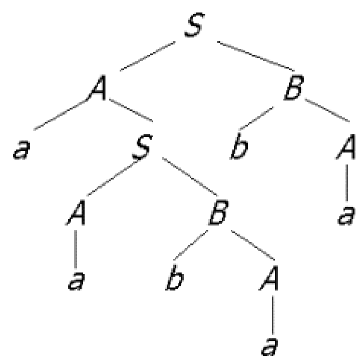
17. Find in the list below the expression that is the contrapositive of A AND (NOT B) → C OR (NOT D). Note: the hypothesis and conclusion of the choices in the list below may have some simple logical rules applied to them, in order to simplify the expressions.

a) (NOT D) AND C → (NOT B) OR A
b) C AND (NOT D) → (NOT B) OR A
c) D AND (NOT C) → B OR (NOT A)
d) B AND (NOT A) → C OR (NOT D)

Answer submitted: **c)**

You have answered the question correctly.

18. Here is a parse tree that uses some unknown grammar G.



Which of the following productions is surely one of those for grammar G?

a) $A \rightarrow a$

b) $S \rightarrow B$

c) $B \rightarrow AS$

d) $S \rightarrow AbA$

Answer submitted: **a)**

You have answered the question correctly.

19. Which of the following strings is NOT in the Kleene closure of the language {011, 10, 110}?

a) 10110011

b) 110011110

c) 01110111

d) 0111010

Answer submitted: **c)**

You have answered the question correctly.

20. Identify in the list below a sentence of length 6 that is generated by the grammar $S \rightarrow (S)S \mid \varepsilon$

a)) ((( ))

b)))) (((

c) ( ) (( ))

d)) ( ) (( )

Answer submitted: **c)**

You have answered the question correctly.

21. Consider the languge L={a}. Which grammar defines L?

a) G_1:S → AB|C, A → b, C → ε
b) G_1:S → AC|a, A → b
c) G_1:S → d|a, A → c|b|ε
d) G_1:S → AB|a|b, A → b

Answer submitted: **b)**

You have answered the question correctly.

22. h is a homomorphism from the alphabet {a,b,c} to {0,1}. If h(a) = 01, h(b) = 0, and h(c) = 10, which of the following strings is in h^{-1}(010010)?

a) *bcab*
b) *baab*
c) *bacb*
d) *cbcb*

Answer submitted: **a)**

You have answered the question correctly.

23. We can represent questions about context-free languages and regular languages by choosing a standard encoding for context-free grammars (CFG's) and another for regular expressions (RE's), and phrasing the question as recognition of the codes for grammars and/or regular expressions such that their languages have certain properties. Some sets of codes are decidable, while others are not.

In what follows, you may assume that G and H are context-free grammars with terminal alphabet {0,1}, and R is a regular expression using symbols 0 and 1 only. You may assume that the problem "Is L(G) = (0+1)*?", that is, the problem of recognizing all and only the codes for CFG's G whose language is all strings of 0's and 1's, is undecidable.

There are certain other problems about CFG's and RE's that are decidable, using well-known algorithms. For example, we can test if L(G) is empty by finding the pumping-lemma constant n for G, and checking whether or not there is a string of length n or less in L(G). It is not possible that the shortest string in L(G) is longer than n, because the pumping lemma lets us remove at least one symbol from a string that long and find a shorter string in L(G).

You should try to determine which of the following problems are decidable, and which are undecidable:

- Is Comp(L(G)) equal to (0+1)*? [Comp(L) is the complement of language L with respect to the alphabet {0,1}.]
- Is Comp(L(G)) empty?
- Is L(G) intersect L(H) equal to (0+1)*?
- Is L(G) union L(H) equal to (0+1)*?
- Is L(G) finite?
- Is L(G) contained in L(H)?
- Is L(G) = L(H)?
- Is L(G) = L(R)?
- Is L(G) contained in L(R)?
- Is L(R) contained in L(G)?

Then, identify the true statement from the list below:

a) "Is L(G) = L(R)?" is decidable.
b) "Is L(H) contained in L(G)?" is decidable.

c) "Is L(G) finite?" is decidable.

d) "Is Comp(L(G)) equal to (0+1)*?" is undecidable.

Answer submitted: **c)**

You have answered the question correctly.

24. Here are eight simple grammars, each of which generates an infinite language of strings. These strings tend to look like alternating *a*'s and *b*'s, although there are some exceptions, and not all grammars generate all such strings.

1. S → abS | ab
2. S → SS | ab
3. S → aB; B → bS | a
4. S → aB; B → bS | b
5. S → aB; B → bS | ab
6. S → aB | b; B → bS
7. S → aB | a; B → bS
8. S → aB | ab; B → bS

The initial symbol is S in all cases. Determine the language of each of these grammars. Then, find, in the list below, the pair of grammars that define the same language.

a) G1: S → abS, S → ab
G2: S → SS, S → ab

b) G1: S → aB, B → bS, B → ab
G2: S → SS, S → ab

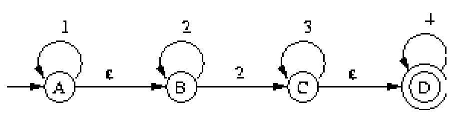
c) G1: S → aB, B → bS, B → b
G2: S → aB, B → bS, S → b

d) G1: S → abS, S → ab
G2: S → aB, B → bS, B → ab

Answer submitted: **a)**

You have answered the question correctly.

25. Here is a nondeterministic finite automaton with epsilon-transitions:



Suppose we use the extended subset construction from Section 2.5.5 (p. 77) to convert this epsilon-NFA to a deterministic finite automaton with a dead state, with all transitions defined, and with no state that is inaccessible from the start state. Which of the following would be a transition of the DFA?

Note: we use S-x->T to say that the DFA has a transition on input x from state S to state T.

a) {C,D}-4->{C,D}

b) {D}-3->{}

c) {A,B}-1->{A}

d) {B,C,D}-2->{B,C}

Answer submitted: **b)**

You have answered the question correctly.