CS 310 (sec 20) - Winter 2003 - Final Exam (solutions)

SOLUTIONS

1. (Logic) Use truth tables to prove the following logical equivalences:

(a) $p \vee q \equiv (p \to \overline{p}) \to \overline{(q \to \overline{q})}$

(b) $p \wedge q \equiv (p \to \overline{q}) \to \overline{(p \to \overline{q})}$

Solution:

(a)

p	q	$p \vee q$	$p \to \overline{p}$	$q \to \overline{q}$	$\overline{(q \to \overline{q})}$	$(p \to \overline{p}) \to \overline{(q \to \overline{q})}$
T	T	T	F	F	T	T
T	F	T	F	T	F	T
F	T	T	T	F	T	T
F	F	F	T	T	F	F

They have the same truth values

(b)

p	q	$p \wedge q$	$p \to \overline{q}$	$\overline{(p \to \overline{q})}$	$(p \to \overline{q}) \to \overline{(p \to \overline{q})}$
T	T	T	F	T	T
T	F	F	T	F	F
F	T	F	T	F	F
F	F	F	T	F	F

They have the same truth values

2. (Relations) Let X be the set of all 4-bit strings (e.g., 0011, 0101, 1000). Define a relation $\mathcal{R}$ on X as $s_1 \,\mathcal{R}\, s_2$ if some substring of s_1 of length 2 is equal to some substring of s_2 of length 2.[1] Examples: $0111 \,\mathcal{R}\, 1010$ (because both 0111 and 1010 contain 01), but $1110 \,\not\mathcal{R}\, 0001$ (because 1110 and 0001 do not have a common substring of length 2). Determine whether $\mathcal{R}$ is: reflexive, symmetric, antisymmetric, transitive, a partial order and/or an equivalence relation.

Solution:

1. Reflexive: YES. Given string $\alpha \in X$, α indeed has some common substring of size 2 with itself, say its substring consisting of the first two bits of α.

2. Symmetric: YES. If $\alpha, \beta \in X$, and α has some substring of size 2 in common with β, also β has that same substring in common with α.

3. Antisymmetric: NO. Two strings in X may be related to each other without being equal, for instance $0111 \,\mathcal{R}\, 1010$ (because both 0111 and 1010 contain 01), but $0111 \neq 1010$.

4. Transitive: NO. Counterexample: $1110 \,\mathcal{R}\, 1100$ (both contain 11) and $1100 \,\mathcal{R}\, 0001$ (both contain 00), however $1110 \,\not\mathcal{R}\, 0001$.

5. Partial order: NO, since the relation is not antisymmetric nor transitive.

6. Equivalence relation: NO, since since the relation is not transitive.

[1]By definition s is a substring of t if there are strings u and v such that $t = usv$.

3. (Algorithms) Let $T(n)$ be the number of times the statement ’`x := x + 1`’ is executed in the pieces of pseudocode below. Select a theta notation for $T(n)$ from among $\Theta(1)$, $\Theta(\log_2(n))$, $\Theta(n)$, $\Theta(n\log_2(n))$, $\Theta(n)$, $\Theta(n^2)$, $\Theta(n^3)$, $\ldots$, $\Theta(2^n)$, $\Theta(n!)$.

```
procedure proc1(n)
  for i := 1 to n do
    for j := 1 to n do
      for k := 1 to n do
        x := x + 1
end proc1
```

```
procedure proc2(n)
  if n = 1 then
    x := x + 1
  else
    begin
      call proc2(n-1) // (two recursive calls
      call proc2(n-1) //  in a row)
    end
end proc2
```

```
procedure proc3(n)
  for i := 1 to n-1 do
    for j := 1 to n-i do
      x := x + 1
end proc3
```

Solution:

1. `proc1(n)`: $T(n) = n^3 = \boxed{\Theta(n^3)}$.
2. `proc2(n)`: $T(1) = 1$, $T(n) = 2T(n-1) = 4T(n-2) = \cdots = 2^nT(1) = 2^n = \boxed{\Theta(2^n)}$.
3. `proc3(n)`: $T(n) = (n-1) + (n-2) + \cdots + 1 = \dfrac{n(n-1)}{2} = \boxed{\Theta(n^2)}$.

4. (Probability) We have two coins. One of them is fair, i.e., the probabilities of head and tails are both equal to 1/2. The other one is loaded, so that the probability of getting tails after tossing it is 1/3 and the probability of head is 2/3. We choose one of the coins at random (with probability 1/2) and toss it.

1. What is the probability of getting "tails".
2. Assume we get "tails". What is the probability that the coin we just tossed is the loaded one?

Give the probabilities as fractions (no decimals!).

Solution:

Let F represent fair, L loaded, H head, T tails.

1. Probability of "tails":

$$P(T) = P(T \mid F) \cdot P(F) + P(T \mid L) \cdot P(L) = \frac{1}{2} \cdot \frac{1}{2} + \frac{1}{3} \cdot \frac{1}{2} = \boxed{\frac{5}{12}}.$$

2. Bayes' theorem:

$$P(L \mid T) = \frac{P(T \mid L) \cdot P(L)}{P(T)} = \frac{\frac{1}{3} \cdot \frac{1}{2}}{\frac{5}{12}} = \boxed{\frac{2}{5}}.$$

5. (Combinatorics)

1. How many strings can be formed by ordering the letters of NORTHWESTERN so that all E's appear between the 2 N's?
2. Find the coefficient of x^3yz^4 in the expansion of $(2x+y+z)^8$.

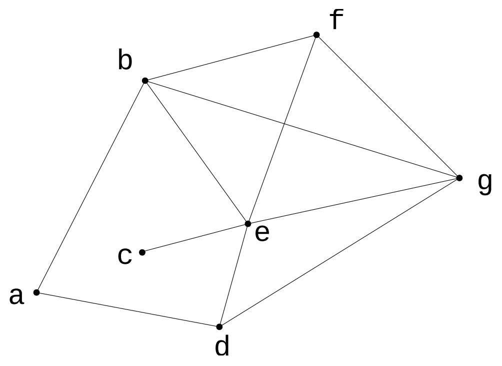
Solution:

1. The word NORTHWESTERN has 12 letters: 2 E's, 1 H, 2 N's, 1 O, 2 R's, 1 S, 2 T's, and 1 W. First we choose 4 places for the 2 E's and the 2 N's, which can be done in $\binom{12}{4}$ ways. Then we place the 2 E's and the 2 N's so that the E's are between the N's; this can be done in just 1 way: $\dots$N$\dots$E$\dots$E$\dots$N$\dots$. Then we place the remaining 8 letters in the remaining places, which can be done in $P(8;2,2,1,1,1,1,1,1) = \dfrac{8!}{2!2!}$ ways. So the final answer is $\boxed{\binom{12}{4}\dfrac{8!}{2!2!} = 4989600}$.

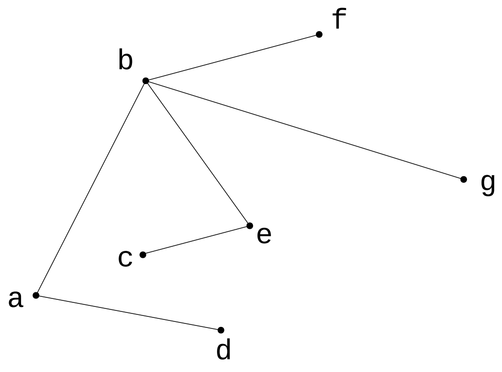
2. $\boxed{\dfrac{2^3 \cdot 8!}{3!1!4!} = 2240}$.

6. (Graphs) In the following graph with its vertices in alphabetical order find a spanning tree using

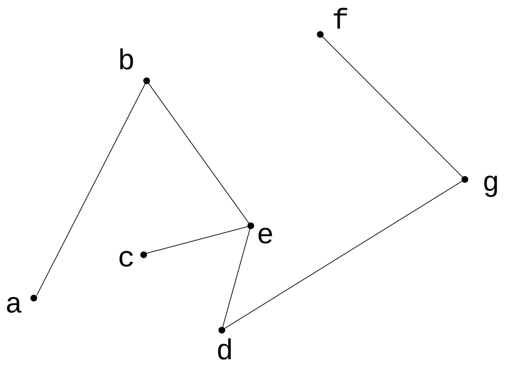
1. Breadth-first search.
2. Depth-first search.



Solution:



Breadth−first



Depth−first

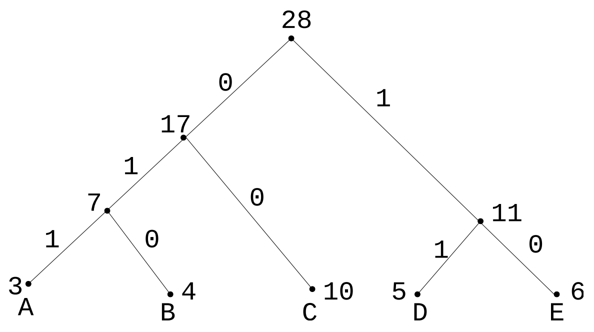
7. (Trees) Construct an optimal Huffman code for the set of letters in the following table (to reduce the number of possible answers always label "1" the edge going from a vertex to its child of lowest frequency and "0" the edge that goes to the child with highest frequency):

letter	frequency
A	3
B	4
C	10
D	5
E	6

Find the average length of bit strings encoding 28-letter words with the Huffman code.

Solution:

The following is the tree corresponding to the optimal Huffman code:



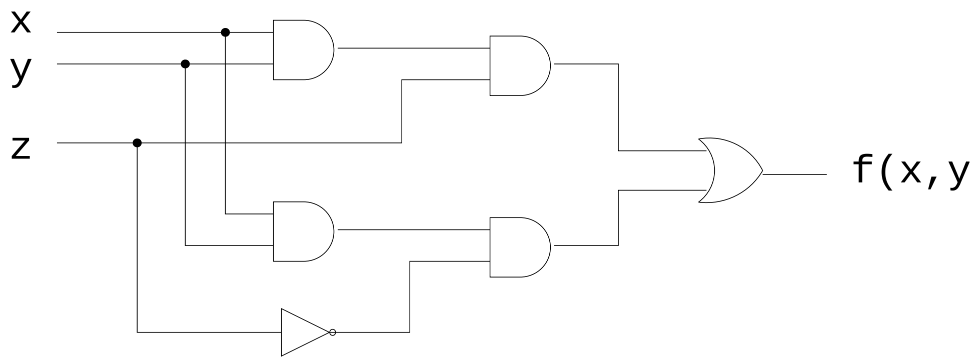
The resulting code is as follows:

letter	code
A	011
B	010
C	00
D	11
E	10

The average length of bit strings encoding 28-letter words is

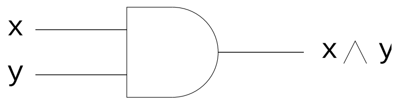
$$3 \cdot 3 + 4 \cdot 3 + 10 \cdot 2 + 5 \cdot 2 + 6 \cdot 2 = \boxed{63} \qquad (= 2.25 \text{ per bit})\,.$$

8. (Combinatorial Circuits and Boolean Algebras) Write the output $f(x, y, z)$ of the following combinatorial circuit as a Boolean expression involving x, y and z. Simplify that Boolean expression. Design an equivalent simpler circuit based on the simplified expression using the minimum possible number of gates.



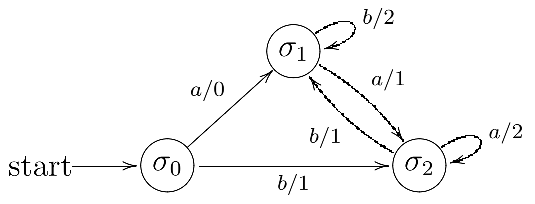
Solution:

$$f(x, y, z) = (x \wedge y \wedge z) \vee (x \wedge y \wedge \overline{z}) = x \wedge y\,.$$



9. (Finite-State Machines) In the following questions the input is always a string over $\{a, b\}$.

(a) Find the output string of the following finite-state machine for the input *ababbbaba*:

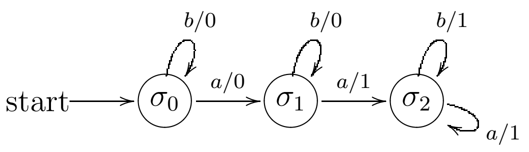


(b) Design a finite-state machine that outputs 1 if a string with two or more a's (an any number of b's) is input; otherwise, outputs 0; e.g., input *babbababb* would produce output 000011111 (it starts outputting 1's right after the second a has been input).

Solution:

(a) 021122111.

(b)



10. (Languages) Let G be the grammar with non terminal symbols $\{E, T, F\}$, terminal symbol $\{a, +, *, (,)\}$, productions

$$E \to T\,, \quad E \to E + T,$$

$$T \to F\,, \quad T \to T * F,$$

$$F \to (E)\,, \quad F \to a,$$

and starting symbol E.

1. Find a derivation for the following string:

$$a * a + (a + a * a)$$

2. Prove that the following string is not in the language $L(G)$ associated to the given grammar:

$$a + a * (a + a * (a + a)$$

Solution:

1. There are many ways to derive the given string, the following is one:

$$E \Rightarrow E+T \Rightarrow T+T \Rightarrow T*F+T \Rightarrow F*F+T \Rightarrow a*F+T \Rightarrow a*a+T$$
$$\Rightarrow a*a+F \Rightarrow a*a+(E) \Rightarrow a*a+(E+T) \Rightarrow a*a+(T+T)$$
$$\Rightarrow a*a+(F+T) \Rightarrow a*a+(a+T) \Rightarrow a*a+(a+T*F) \Rightarrow a*a+(a+F*F)$$
$$\Rightarrow a*a+(a+a*F) \Rightarrow a*a+(a+a*a)$$

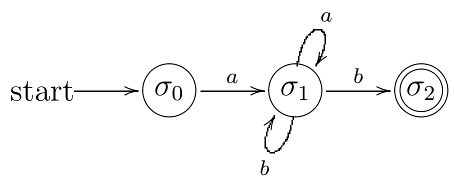
2. Parentheses can be produced only by using the rule $F \to (E)$, so each time we produce a left parenthesis we also produce a right parenthesis, hence in the final string the number of left parentheses must be equal to the number of right parentheses. However the given string has two left parentheses and only one right parenthesis.

11. (Extra-Credit)

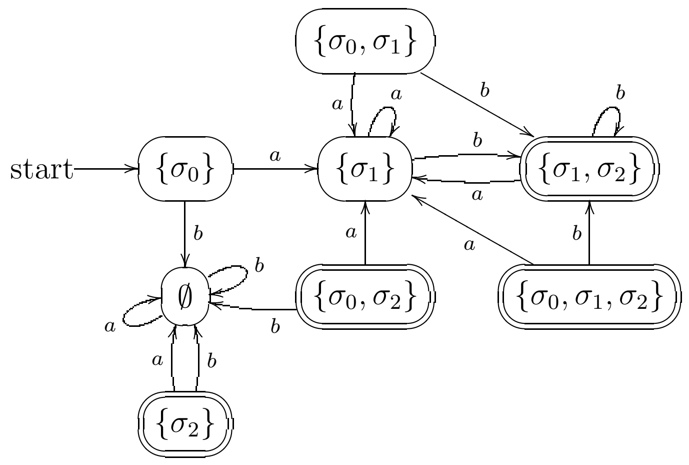
1. Design a finite-state automaton (by giving its transition diagram) that recognizes the strings represented by the regular expression $a(a+b)^*b$.
2. Is your finite-state automaton deterministic or non-deterministic? If it is non-deterministic design an equivalent deterministic automaton (and simplify it by removing unreachable states).

Solution:

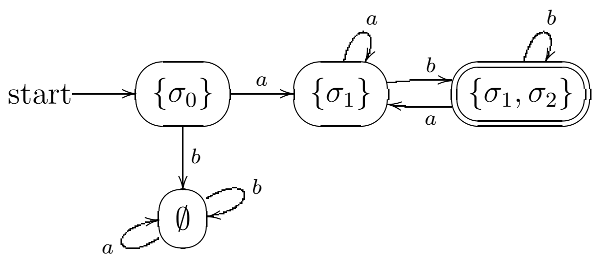
1.



2.



After removing unreachable states we get the following:


a
b
start
$\{\sigma_0\}$
a
$\{\sigma_1\}$
b
$\{\sigma_1, \sigma_2\}$
a
b
b
$\emptyset$
a