Mid-Semester Test # 1 solutions

1) Design a DFA that accepts the set of strings over \{0,1\} such that every block of three consecutive symbols contains at least one 1. (Strings of length 0, 1 and 2 should to be accepted.) To get full credit, your DFA should have at most 8 states.

*The following is the optimal DFA for this language:*

![DFA Diagram](image)

2) Find a succinct (and natural) English description for the set of strings (over the alphabet \{0,1\}) generated by the regular expression \((0 \cup 10^*)10^*\). Convert this regular expression into an NFA using the algorithm presented in class.

\[ L(r) = \{ w \mid w \text{ has an odd number of 1's} \} \]

*The NFA equivalent to the given regular expression is as shown below. Note that the algorithm from the text was not followed completely. Some simplifications were made to reduce the size of the resulting NFA.*
3) (a) Design an NFA for the set of strings \( w \) over \( \{0, 1, 2\} \) such that \( w \) omits at least one of the three symbols. Example: 01101, 212121, 0000, 000222, 111 etc. are in the language. Strings like 201, 220102, 110022 etc. are not in the language. For full-credit, your NFA should have at most 4 states.

(b) Express the DFA shown below using the 5-tuple notation \(<Q, \Sigma, \delta, s, F>\). (Describe each of the components of the 5-tuple using a set or tabular notation.)

\[ Q = \{q0, q1, q2\} \]

Start state = \( q0 \)
Final states \( F = \{ q1 \} \)

Transition function \( \delta \):
\[
\begin{align*}
\delta(q0, 0) &= q2, \quad \delta(q0, 1) = q0 \\
\delta(q1, 0) &= q1, \quad \delta(q1, 1) = q1 \\
\delta(q2, 0) &= q2, \quad \delta(q2, 1) = q1
\end{align*}
\]

4) (a) State the negation of the following statement: For every positive integer \( x \), there is a prime number \( y \) such that \( y > x \).

There is a positive integer \( x \) such that for all prime numbers \( y \), \( x > y \).

(b) State the converse and the contrapositive of the following statement: if a positive integer \( N \) can be written as a sum of \( k \) consecutive integers for some \( k > 0 \) then \( N \) is not a power of 2.

Converse: If a positive integer \( N \) is not a power of 2, then \( N \) can be written as a sum of \( k \) consecutive integers for some \( k > 0 \).

Contrapositive: If a positive integer \( N \) is a power of 2, then \( N \) can't be written as a sum of \( k \) consecutive integers for some \( k > 0 \).

(c) Show by induction on \( N \) that \( 1 + 2 + 2^2 + \ldots + 2^{N-1} < 2^N \).

Proof: The base case \( N = 1 \). LHS = 1, RHS = 2, so LHS < RHS is obvious/
Induction step: Assume the claim is true for \( N = k \), and show it is true for \( N = k + 1 \).

By induction hypothesis, \( 1 + 2 + 2^2 + \ldots + 2^{k-1} < 2^k \)

We need to show \( 1 + 2 + 2^2 + \ldots + 2^k < 2^{k+1} \)

\[
LHS = 1 + 2 + 2^2 + \ldots + 2^k = 1 + 2 + 2^2 + \ldots + 2^{k-1} + 2^k < 2^k + 2^k \quad \text{(by I.H.)}
\]

\[
= 2^{k+1} = RHS
\]

(d) If \( R \) is a reflexive and a transitive relation, is \( R \) also a symmetric relation?
Justify your answer.

Not necessarily. Counterexample: \( \{(a, a), (b,b), (c, c), (a, b)\} \)

5) Convert the following NFA into an equivalent DFA.
6) Write a regular expression for the set of strings over \{0, 1\} that do not have 101 as a substring. Show all the steps clearly.

There are at least two ways to solve this problem. The first one is to create a DFA for the language and convert it to R.E. using the conversion algorithm.

The DFA for the language (after removing the dead state) is as shown below:
Converting this into R.E. (by using some short-cuts) is:

\[(0 + 11*00)^*(\varepsilon + 11^* + 11^*0)\]

7) (a) Design a DFA to accept the set of strings \((aa \cup aaaa)^*\) over the alphabet \(\{a\}\). For full credit, the DFA should have at most 3 states.

(b) All strings except \(\{a, aaaaa, aaaaaaa\}\).

8) Consider a string over the alphabet \([0,0],[0,1],[1,0],[1,1]\). We can view the string as having two components – the first one is obtained by concatenating the bits of the first component and the second one by concatenating the second component bits. For example, the two components of \([0,1][1,0][1,1]\) are 011 and 101. Next we define the notion of right rotation with an example. The string 1110 is a right rotation of 0111. Define a language \(L = \{x \mid \text{second component of } x \text{ is a right rotation of the first component}\}\). For example, strings such as \([0,1][1,1][1,1][1,0]\) and \([1,0][0,1][1,1][1,1]\) are in \(L\). But \([1,1][1,0][0,0]\) is not in \(L\). Design a DFA for the \(L\).

DFA is shown below. The missing transitions are to a dead state (that is also not shown).