Michael Wang
CS 181 HW 2

2d) {0x, 0y, 0z, 1x, 1y, 1z}

2e) {}

1a) $L_{Reflection}$ is not equal to L_{PAL} because L_{PAL} recognizes strings of length 1 while $L_{Reflection}$ does not (otherwise, both languages recognize palindromes).

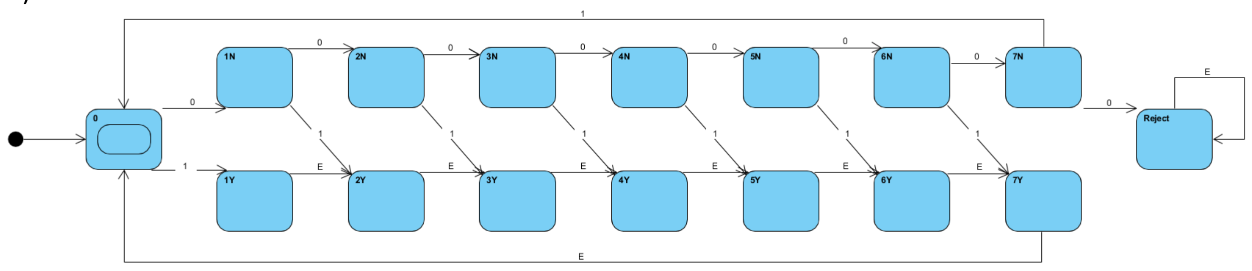
1b) L_1 recognizes strings that are the concatenation of some string from Σ^* and another from Σ^{*R}. The two strings need not be related in any way, for example the first string could be 1 and the second 0, which when concatenated clearly do not belong to L_{PAL} which only recognizes palindromes. Thus, L_1 and L_{PAL} are not equivalent.

2a) A k-ary directed rooted tree of height n can have a minimum of n+1 nodes and n edges. Such a tree is constructed by connecting a single child to the root node then connecting a single child to the most recently added node until there are n+1 nodes. The last node has height n and n edges are created. No nodes can be removed without decreasing the height of the tree and no edges can be removed without disconnecting the tree, thus the minimum number of each has been achieved.

2b) A k-ary directed rooted tree of height n can have a maximum of $\sum_{i=1}^{n+1} k^{i-1} = \frac{1-k^{n+1}}{1-k}$ nodes and that number of edges minus 1. Such a tree is constructed by recursively adding k children to the root node and then adding k children to all leaf nodes. To prove such a tree is optimal, assume the base case of n = 1. The maximum number of nodes is achieved by connecting k children to the root node, resulting in $\sum_{i=1}^{n+1} k^{i-1} = 1 + k$ nodes and 1 fewer edges. Assume a tree up to height n-1 have been constructed this way totaling $\sum_{i=1}^{n} k^{i-1}$ nodes. At this point there are k^{n-1} leaf nodes. Maximizing the number of nodes in a tree of height n would clearly require adding k children to each of the leaf nodes, resulting in in k^n new nodes and a total of $\sum_{i=1}^{n+1} k^{i-1}$ nodes. Thus the base case and inductive cases are both true.

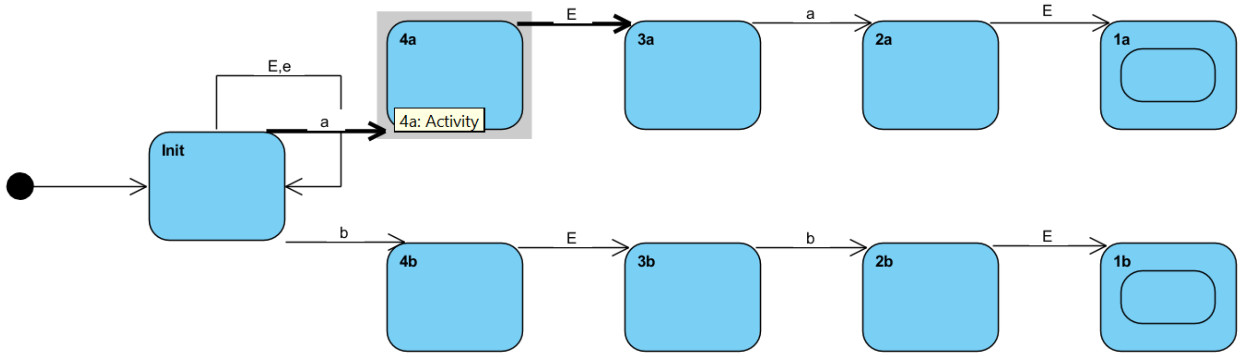
3) The given DFA recognizes languages that contain at least one b and at least one a following the first b.

4)



In this DFA, the number of symbols modulo eight is kept track of. Additionally, the DFA records whether a one has appeared in the byte so far. If the byte completes without having a 1, the reject state is reached. Otherwise the cycle begins anew. No states go to the next No state when encountering a 0 or the next Yes state when encountering a 1. Yes states go to the next state no matter what symbol they encounter. It was assumed that the empty string was recognized. If not, the DFA would require an additional state to which 7N and 7Y transition when encountering a 1.

5)



In this NFA, there are two possible branches with the first remembering that the fourth to last symbol is a and the other remembering that the fourth to last symbol is b. Both branches allow the third and last symbol to be anything while the second to last must match the fourth.

6)
(Σ*1* Σ *0* Σ*) U (Σ*0* Σ *1* Σ*)
Perhaps there is a shorter way of representing this language since the above regular expression is rather verbose. The first half states that there must be at least one 1 anywhere in the string followed by at least one 0 anywhere in the rest of the string. The second half states that there must be a t least one 0 anywhere in the string followed by at least one 1 in the rest of the string. Taking the union of these sets provides us with the desired language as any string containing a one and zero falls into at least one of the two and all strings recognized by the regular expression contain a 1 and a 0.