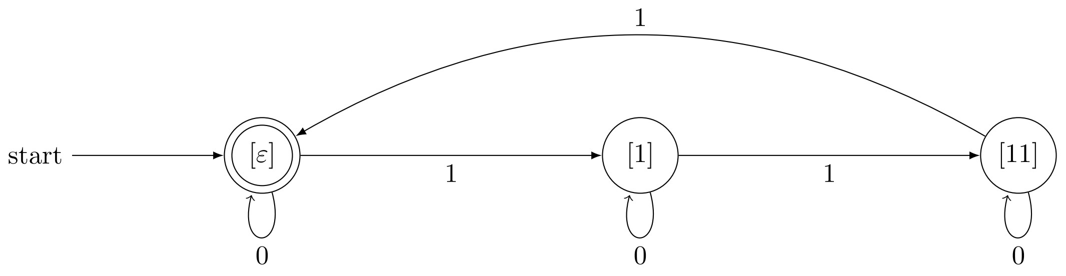
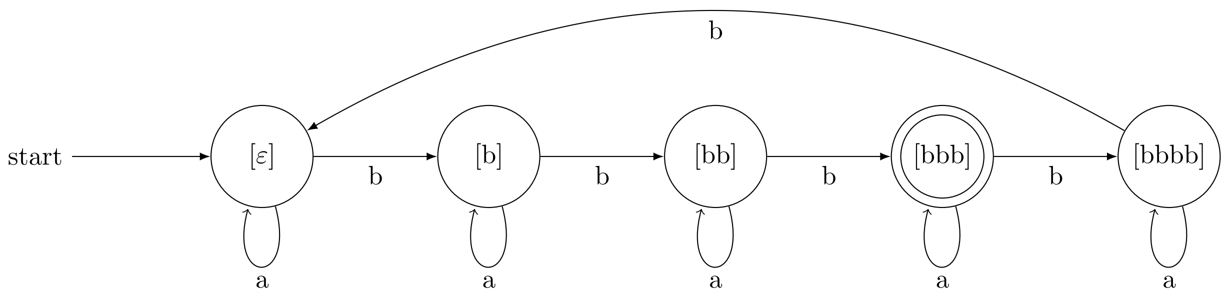
Problem 1 Solution



Problem 2a Solution (5 points)



Problem 2b Solution (5 points)

(one possible solution)

$b \not\equiv_L$ *bb*,*bbb*,*bbbb* $\cdots$ since *aa* distinguishes *b* from *bb*, *bbb*, *bbbb*, and so on.

Similarly, $bb \not\equiv_L$ *bbb*,*bbbb*,*bbbbb* $\cdots$ since *aaaa* would distinguish it from the rest.

This continues and so each different length string of *b*'s is pairwise indistinguishable from the rest. Therefore, there are an infinite number of equivalence classes and so the language is not regular by the Myhill-Nerode Theorem.

Problem 2c Solution (5 points)

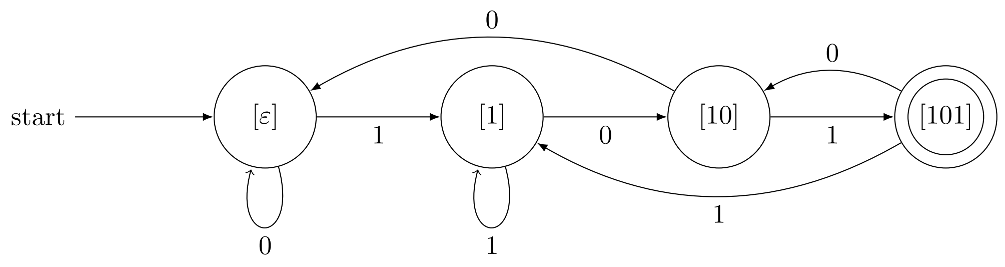
(one possible solution)

$1\oplus \not\equiv_L 2\oplus, 3\oplus, 4\oplus \cdots$ since $1 \ominus 2$ distinguishes $1\oplus$ from the rest.

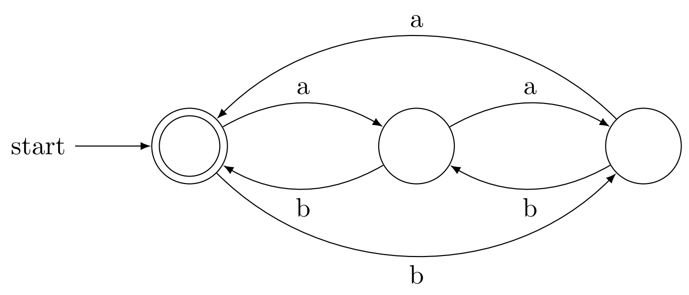
Similarly, $2\oplus \not\equiv_L 3\oplus, 4\oplus, 5\oplus \cdots$ since $1 \ominus 3$ distinguishes $2\oplus$ from the rest.

This continues and so there are an infinite number of equivalence classes and so the language is not regular by the Myhill-Nerode Theorem.

Problem 2d Solution (5 points)



Problem 3

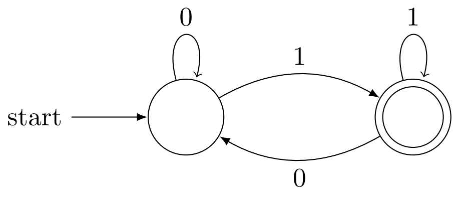


Problem 4 Part 1: DFA Solution

(Possible solution)

Induction

Base case: If $k = 1$ this DFA solves it and has $2^k = 2^1$ states.



Inductive hypothesis: Assume $\exists k \in \mathbb{Z}^+$ such that $\forall k' < k$ the number of states needed for $L_{k'}$ is $O(2^{k'})$.

Now consider L_{k+1}. By the inductive hypothesis there is a DFA with $\Omega(2^k)$ states for L_k. We can modify this DFA to decide L_{k+1}: For each accepting state of this DFA, change it to an unaccepting state, remove its outgoing transitions, and create two new accepting states for it to transition to (one for a 1 and one for a 0). Since the old accepting state accepted strings with 1 in the k^{th}-to-last position, having one more bit at the end transitions us to these new accepting states which now exactly correspond to having a 1 in the $(k+1)^{st}$-to-last position.

To finish this we need to know what transitions come out of the new accepting states. Since the DFA could have been built up inductively with this process from the base case, it can be seen that each accepting state is a leaf node of tree and represents all strings that end in the $k+1$ characters it took for a direct path from the start state to that leaf node. Moreover, each state in the tree can be seen as representing all strings that end in the characters it took to get there in a direct path from the start state. Finally, to create the transitions from the accepting states, we simply read in the next bit (which will shift what the last $k+1$ characters are) and, since the shift shoves our $(k+1)^{st}$ 1 off the end, we see what the next furthest 1 there is from the end in the last $k+1$ characters; this 1 and the remaining characters is represented in one of the internal nodes of the tree and we should jump there.

This resulting DFA will decide L_{k+1} and has $O(2^{k+1})$ states.

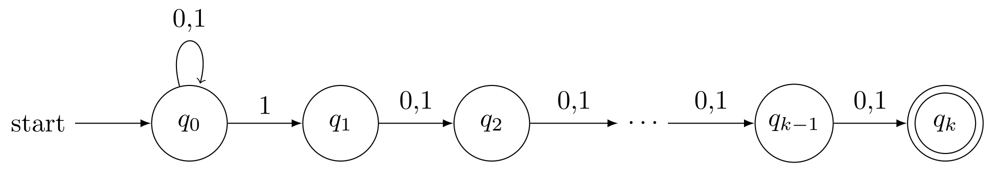
This however is more constructive than necessary and shows a DFA with $O(2^k)$ states to solve the problem rather than proving that $\Omega(2^k)$ are needed for any DFA. To show this, we must show that our states are distinguishable so that each of the $O(2^k)$ states are needed:

We've already stated that each node represents the equivalence class of the string it took to get to that node as a direct path down the tree from the start state. Now we just must show that each of these strings is distinguishable. In fact, to make things easier, we will prepend each of these strings with $(k - depth)$ zeroes based on the depth of each string's node in the tree. Each of these new k-length strings now represents a node in our exponential tree and we will argue that each pair is distinguishable.

For any two of these strings, they will be distinct and so we can find a position in which they differ (i.e. there is a position, say the n^{th} to last position, where one string has a 1 while the other has a 0). Append $(k - n)$ zeroes to both strings. The string with 1 in the n^{th} to last position will be accepted now while the other will not by construction.

This shows that each of these strings and states are distinguishable and so all $\Omega(2^k)$ of these states are needed for our DFA.

Problem 4 Part 2: NFA Solution



This NFA decides L_k and has obviously $O(k)$ states.

Problem 4 Part 3: Regular Expression Solution

The regular expression $(0 \cup 1)^* 1 (0 \cup 1)^k$, where the k^{th} power of an expression is shorthand for concatenation k times, describes L_k and, when expanded, is length $O(k)$.