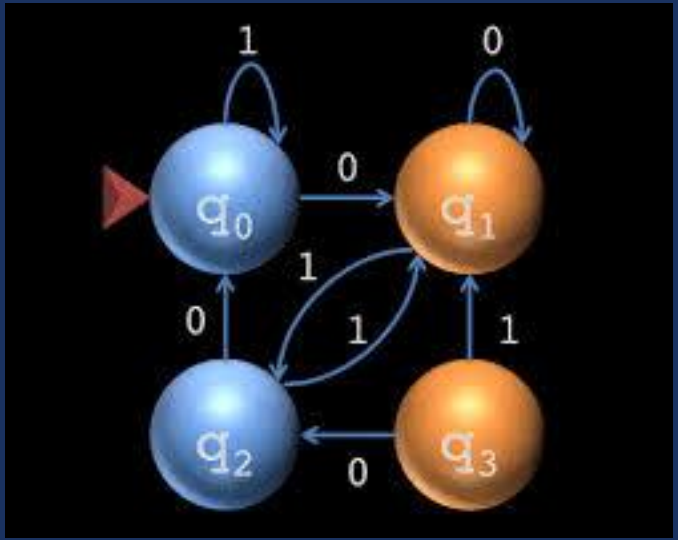
AUTOMATA WITH OUTPUTS

MYHILL NERODE THEOREM – AKA (TABLE FILLING ALGORITHM)



Syed Faisal Ali
faisal.ali@nu.edu.pk
Computer Science D, E & F

SPRING 2025

MYHILL-NERODE THEOREM

Definition:

- A language $\mathbf{L} \subseteq \boldsymbol{\Sigma}^*$ is **regular if and only if** the number of equivalence classes of the **relation ≡L** is **finite**.

What is ≡L (Myhill-Nerode Equivalence)?

- Two strings x and y in Σ^* are equivalent with respect to L (written as x ≡L y) if:

  For every string $z \in \Sigma^*$, the string $xz \in L \Leftrightarrow yz \in L$.

That means:

- x and y are indistinguishable by the language L.
- If you add the same suffix z to both and they always result in the same membership decision (either both are in L or both are not in L), then they are equivalent.

DISTINGUISHABLE AND INDISTINGUISHABLE STATES

Distinguishable States

- Two states are distinguishable if:
- There exists at least one input string that leads one state to accept and the other to reject.
- They behave differently for some string.

Indistinguishable States

- Two states are indistinguishable if:
- For every input string, both states either accept or reject the string.

STEPS

Step 1: Construct the Distinguishability Table

- Create a triangular table (not including diagonal) where each cell represents a pair of states (p, q) with $p < q$.
- You will mark pairs that are distinguishable.

Step 2: Mark Final/Non-final State Pairs

- If one of the states is final and the other is non-final, mark the pair as distinguishable (×).
- These cannot be equivalent because they accept different languages.

STEPS

Step 3: Iteratively Mark More Pairs

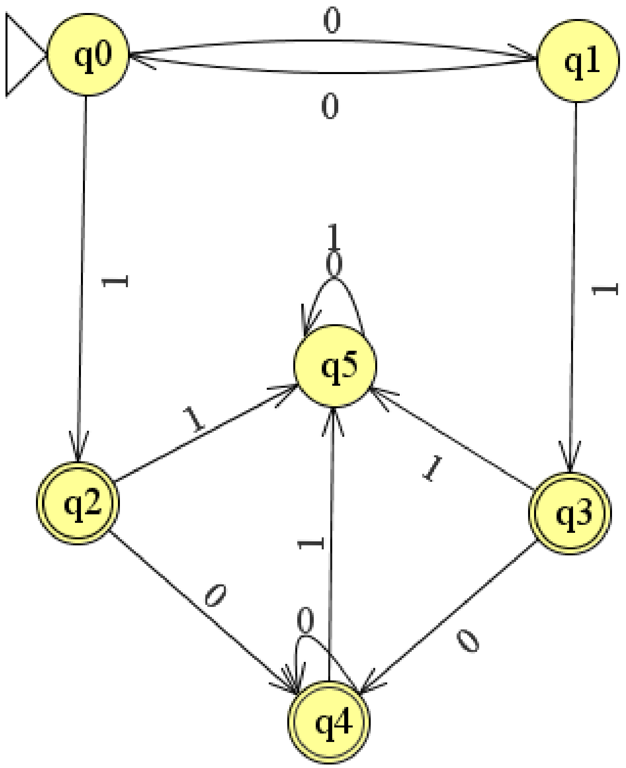
- For each unmarked pair (p, q):
- For each input symbol a:
- Compute δ(p, a) = p1 and δ(q, a) = q1
- Check the pair (p1, q1):
- If it is already marked, then (p, q) must also be marked.
- Repeat this process until no more new markings occur.

Step 4: Merge Equivalent States

- The unmarked pairs are considered equivalent.
- Merge these states to form a minimized DFA.

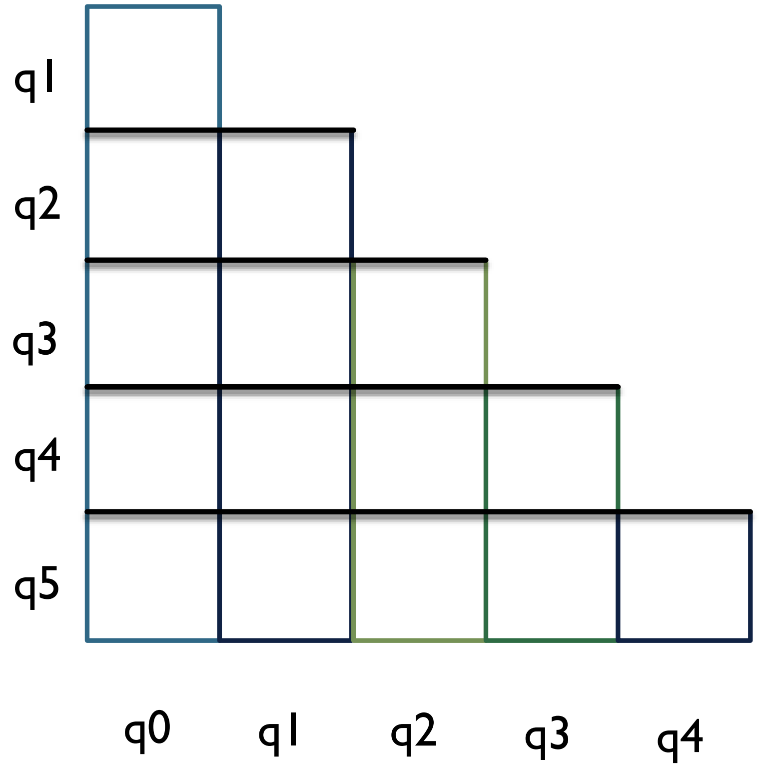
MINIMIZE THE GIVEN DFA

States	0	1
- Q0	Q1	Q2
Q1	Q0	Q3
+ Q2	Q4	Q5
+ Q3	Q4	Q5
+ Q4	Q4	Q5
Q5	Q5	Q5



CONSTRUCT THE TABLE

- Now construct the table as shown in the diagram for the pairs as follows:
- (q0,q1), (q0,q2), (q0,q3), (q0, q4), (q0, q5)
- (q1,q2), (q1,q3), (q1,q4), (q1,q5)
- (q2,q3), (q2,q4), (q2, q5)
- (q3,q4), (q3, q5)
- (q4, q5)
- Now we need to find the distinguish and in-distinguish states for the above mentioned pairs.



MARK DISTINGUISH AND IN-DISTINGUISH STATES IN TABLE

States	Distinguish/ In-distinguish	Marked or Un Marked
(q0,q1)	In-distinguish	Un-Marked
(q0,q2)	Distinguish	Marked
(q0,q3)	Distinguish	Marked
(q0, q4)	Distinguish	Marked
(q0, q5)	In-distinguish	Un-Marked
(q1,q2)	Distinguish	Marked
(q1,q3)	Distinguish	Marked
(q1,q4)	Distinguish	Marked
(q1,q5)	In-distinguish	Un-Marked
(q2,q3)	In-distinguish	Un-Marked
(q2,q4)	In-distinguish	Un-Marked
(q2, q5)	Distinguish	Marked

q1					
q2	★	★			
q3	★	★			
q4	★	★			
q5			★		
	q0	q1	q2	q3	q4

Mark if one of the state is non-final and other is final. Otherwise leave blank.

MARK DISTINGUISH AND IN-DISTINGUISH STATES IN TABLE

States	Distinguish/ In-distinguish	Marked or Un Marked
(q3,q4)	In-distinguish	Un-Marked
(q3, q5)	Distinguish	Marked
(q4, q5)	Distinguish	Marked

q1					
q2	★	★			
q3	★	★			
q4	★	★			
q5			★	★	★
	q0	q1	q2	q3	q4

Mark if one of the state is non-final and other is final. Otherwise leave blank.

- Here we can clearly found that few of the pairs are left blank
- Such as (q0, q1), (q0, q5), (q1, q5), (q2, q3), (q2, q4), and (q3, q4).
- As we have marked pair (one final , other non-final) now we need to perform step 3 for the remaining pairs.

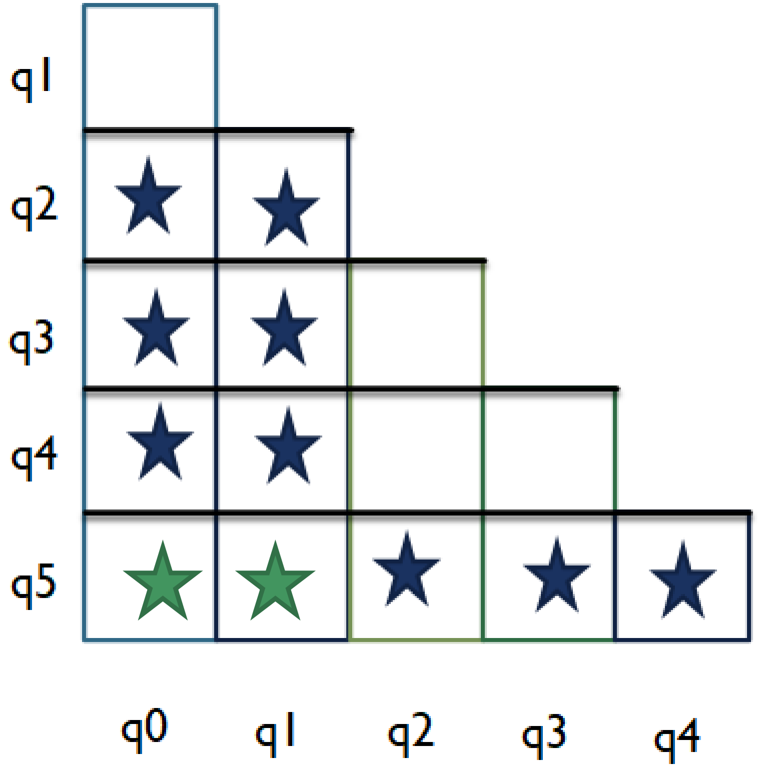
Step 3: Iteratively Mark More Pairs

- For each unmarked pair (p, q):
- For each input symbol a:
- Compute δ(p, a) = p1 and δ(q, a) = q1
- Check the pair (p1, q1):
- If it is already marked, then (p, q) must also be marked.
- Repeat this process until no more new markings occur.

q1					
q2	★	★			
q3	★	★			
q4	★	★			
q5			★	★	★
	q0	q1	q2	q3	q4

STEP 3 : FILLING BASED ON ALREADY MARKED PAIRS

States	At input 0	Check either it is marked already	At input 1	Check either it is marked already	Conclusion
(q0, q1)	(q1, q0)	Not Marked	(q2, q3)	Not Marked	Un-Marked
(q0, q5)	(q1, q5)	Not Marked	(q2,q5)	Marked	Marked
(q1, q5)	(q0, q5)	Marked	No Need	No Need	Marked
(q2, q3)	(q4, q4)	Not Marked	(q5, q5)	Not Marked	Un-Marked
(q2, q4)	(q4, q4)	Not Marked	(q5, q5)	Not Marked	Un-Marked
(q3, q4)	(q4, q4)	Not Marked	(q5, q5)	Not Marked	Un-Marked



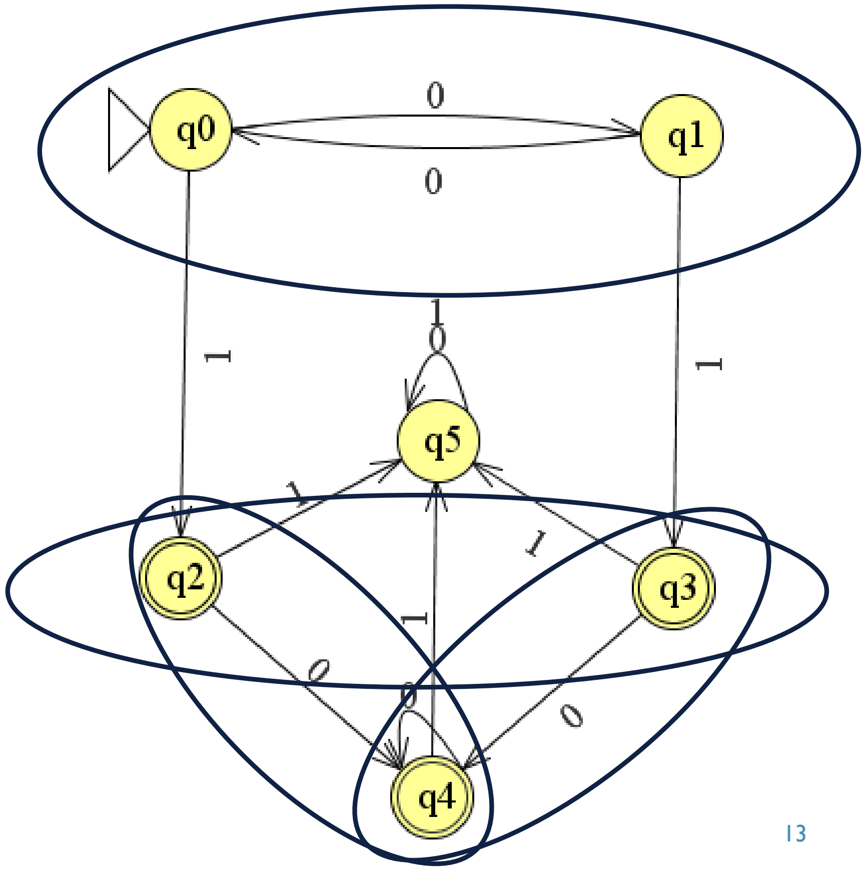
RESULT AFTER PERFORMING STEP 3

- Now as we can see that there are 4 pairs which are not Marked so we need to make the final automata based on these pair of states.
- (q0, q1) combined state
- (q2,q3) combined state
- (q2,q4) combined state
- (q3, q4) combined state

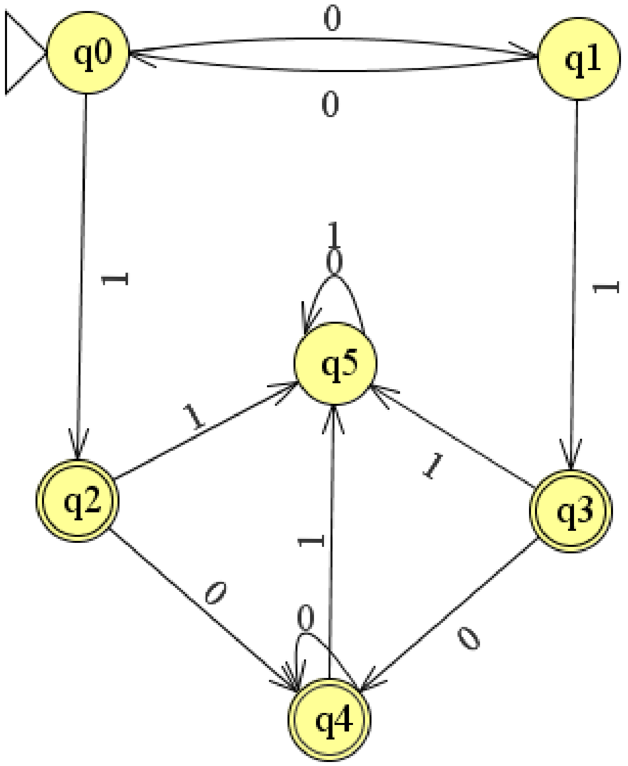
q1					
q2	★	★			
q3	★	★			
q4	★	★			
q5	★	★	★	★	★
	q0	q1	q2	q3	q4

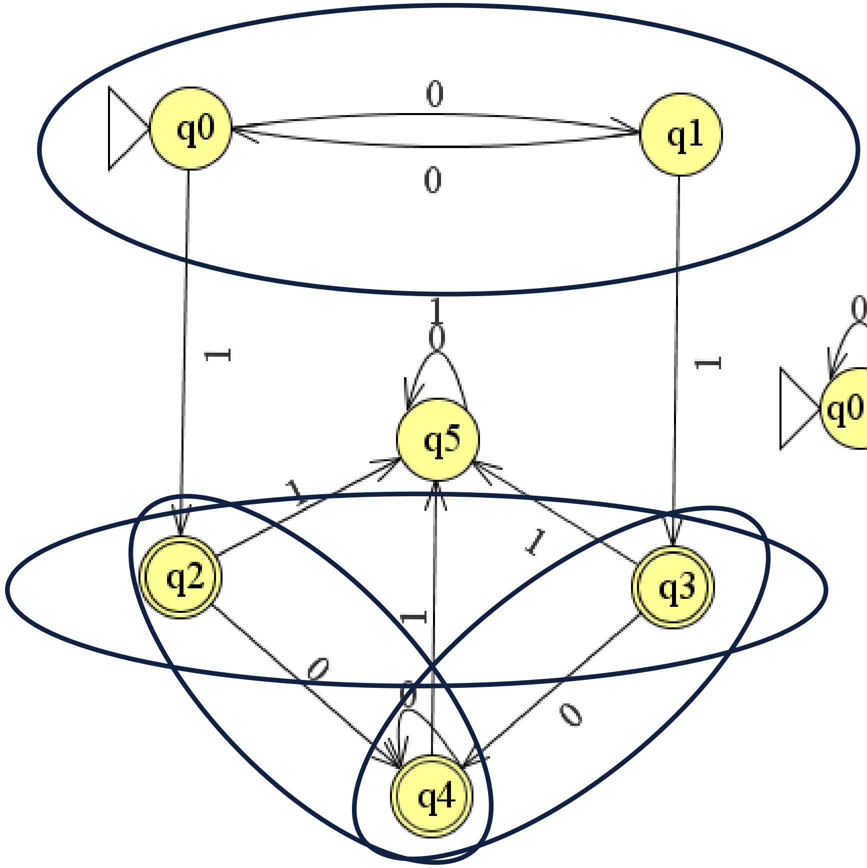
GROUPING THE STATES

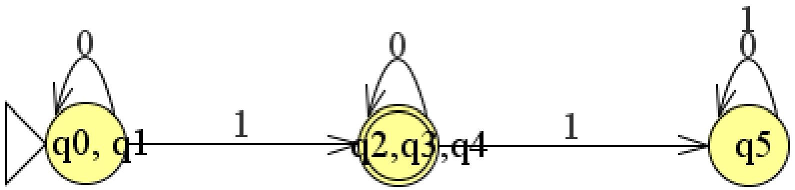
- (q0, q1) combined state
- (q2,q3) combined state
- (q2,q4) combined state
- (q3, q4) combined state
- We can clearly see that q2, q3 and q4 are overlapping states so we combined them into one single state making (q2, q3 and q4) also marked them as final states cause they are final states.
- Remaining one state q5 left alone.
- So now making three states automata



FINAL MINIMIZED AUTOMATA







Minimized Automata

Original Automata

Grouped for Minimized