Finite automata

Irina Prosvirnina

- Closure properties of regular languages
- Pumping lemma

Closure properties of regular languages

<u>Theorem 1</u>

The class of regular languages is closed under union, intersection, subtraction, complementation, concatenation, Kleene closure and reversal.

<u>Proof.</u>

The idea is to build a DFA for the union of two languages by combining the two DFA's into one such that, at each step, the new DFA would keep track of the computation paths of both DFA's.

Proof of theorem 1

- Let $M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ be DFA to accept $L(M_1)$, let $M_2 = (Q_2, \Sigma, \delta_2, q_0, F_2)$ be DFA to accept $L(M_2)$.

Proof of theorem 1

- Consider a product automaton $M = M_1 \times M_2$.

The state set $Q = Q_1 \times Q_2$ is the cross product of the state sets of M_1 and M_2.

The initial state of M is (q_0, q_0).

At each state $\left(q_i, q_j\right)$ in Q, we simulate both computations of M_1 and M_2 in parallel by

$$\delta\left(\left(q_i, q_j\right), a\right) = \left(\delta_1(q_i, a), \delta_2\left(q_j, a\right)\right).$$

Proof of theorem 1

- The set of final states of M:

$$F = (F_1 \times Q_2) \cup (Q_1 \times F_2).$$

From the above description, it is clear that M accepts the union of two languages $L(M_1) \cup L(M_2)$.

Proof of theorem 1

- The above method can also be applied to the problems of finding the intersections or the differences of two languages which are accepted by DFA's.

For the intersection of two regular languages we need to take $F_1 \times F_2$ as F and for the difference of two languages we need to take $F_1 \times (Q_2 - F_2)$ as F.

Proof of theorem 1

- Let $M = (Q, \Sigma, \delta, q_0, F)$ be DFA to accept the language $L(M)$.

Then DFA $(Q, \Sigma, \delta, q_0, Q - F)$ accepts the complementation of the language $L(M)$.

Proof of theorem 1

- Let's show how for given NFA's M_1 and M_2 to construct the NFA accepting the concatenation $L(M_1) \cdot L(M_2)$ of two languages and Kleene closure $L(M_1)^*$ of the language $L(M_1)$.

Let's begin with concatenation of two regular languages.

Proof of theorem 1

- Let $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$ be an NFA to accept $L(M_1)$,

let $M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$ be an NFA to accept $L(M_2)$.

We construct an NFA M such that

$L(M) = L(M_1) \cdot L(M_2)$.

We make a copy of each of M_1 and M_2.

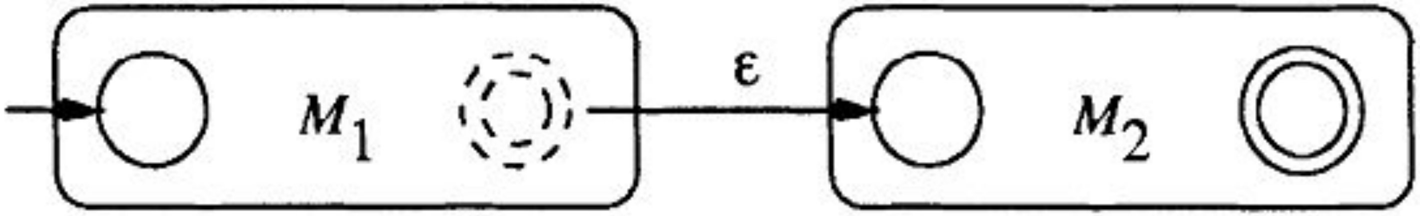
Proof of theorem 1

- Then, we let the initial state q_0^1 of M_1 be the initial state of M and let the set F of the final states of M be equal to F_2.

We also add an ε-move from each state q in F_1 to the initial state q_0^2 of M_2.

Proof of theorem 1

- We construct an NFA M such that $L(M) = L(M_1) \cdot L(M_2)$.
Then, we let the initial state q_0^1 of M_1 be the initial state of M and let the set F of the final states of M be equal to F_2.
We also add an ε-move from each state q in F_1 to the initial state q_0^2 of M_2.



Proof of theorem 1

- Let $M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1)$ be NFA to accept the language $L(M_1)$.

Let's construct an NFA M such that $L(M) = L(M_1)^*$.

We construct M by adding a new initial state s and a unique final state f.

Proof of theorem 1

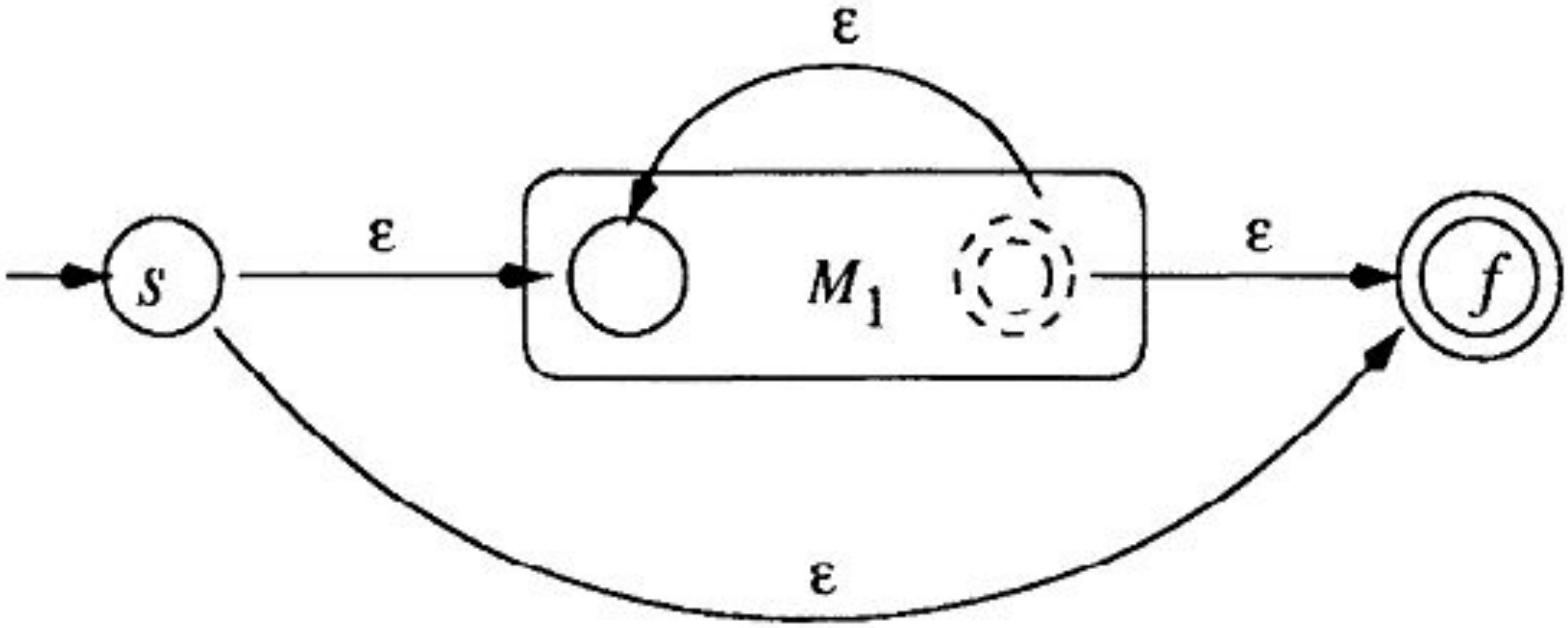
- Then, we add an ε-move from s to the initial state q_0 of M_1 and an ε-move from each $q_i \in F_1$ to the new final state f.

We also add, from each state $q_i \in F_1$, an ε-move to the initial state q_0 of M_1.

Finally, we add an ε-move from the initial state s to the new final state f (so that the empty string ε is accepted).

Proof of theorem 1

Kleene closure of an NFA.



Proof of theorem 1

- If L is regular language then L^R is regular language too:

$\emptyset^R = \emptyset$,

$\{\varepsilon\}^R = \varepsilon$,

$(AB)^R = B^R A^R$,

$(A \cup B)^R = A^R \cup B^R$,

$(A^*)^R = (A^R)^*$.

$\blacksquare$

Closure properties of regular languages

In the following examples a language is given and we show how to construct a DFA or a NFA accepting the language.

Closure properties of regular languages

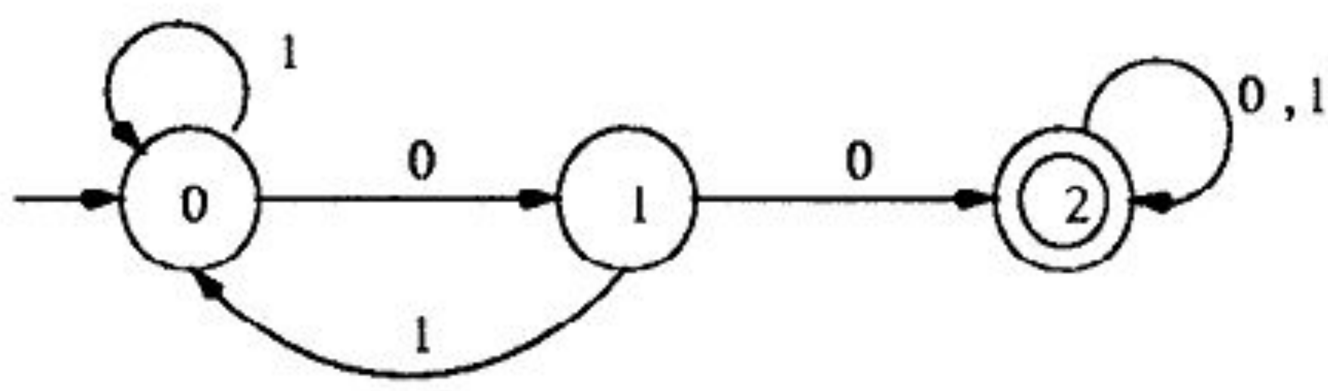
Example 1

The set of all binary strings having a substring 00 or ending with 01.

This language is the union of two languges $(0+1)^*00(0+1)^*$ and $(0+1)^*01$.

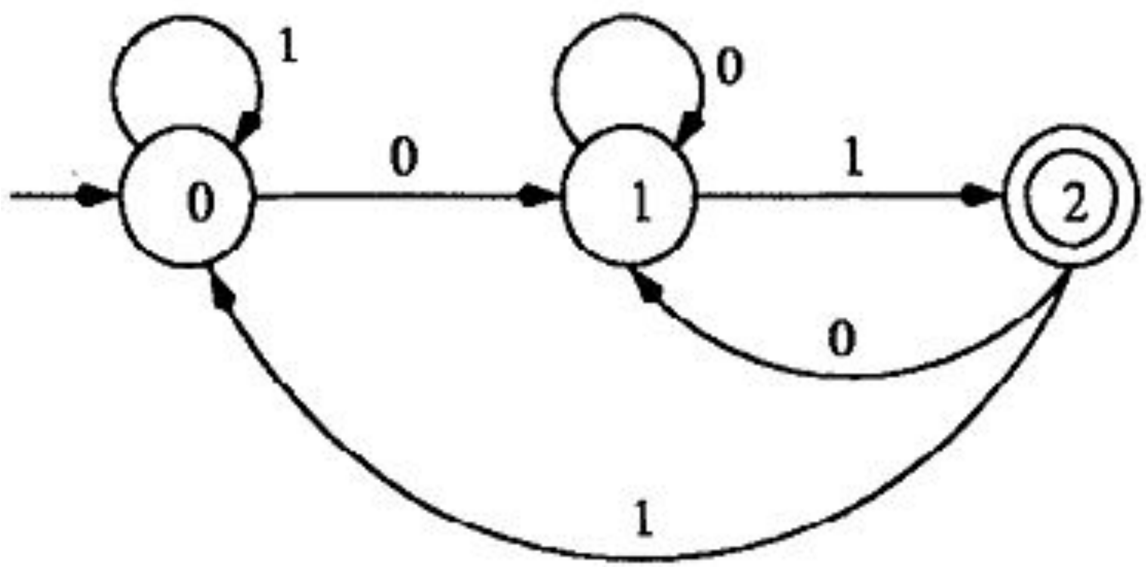
Solution of example 1

- DFA to accept $(0+1)^*00(0+1)^*$.



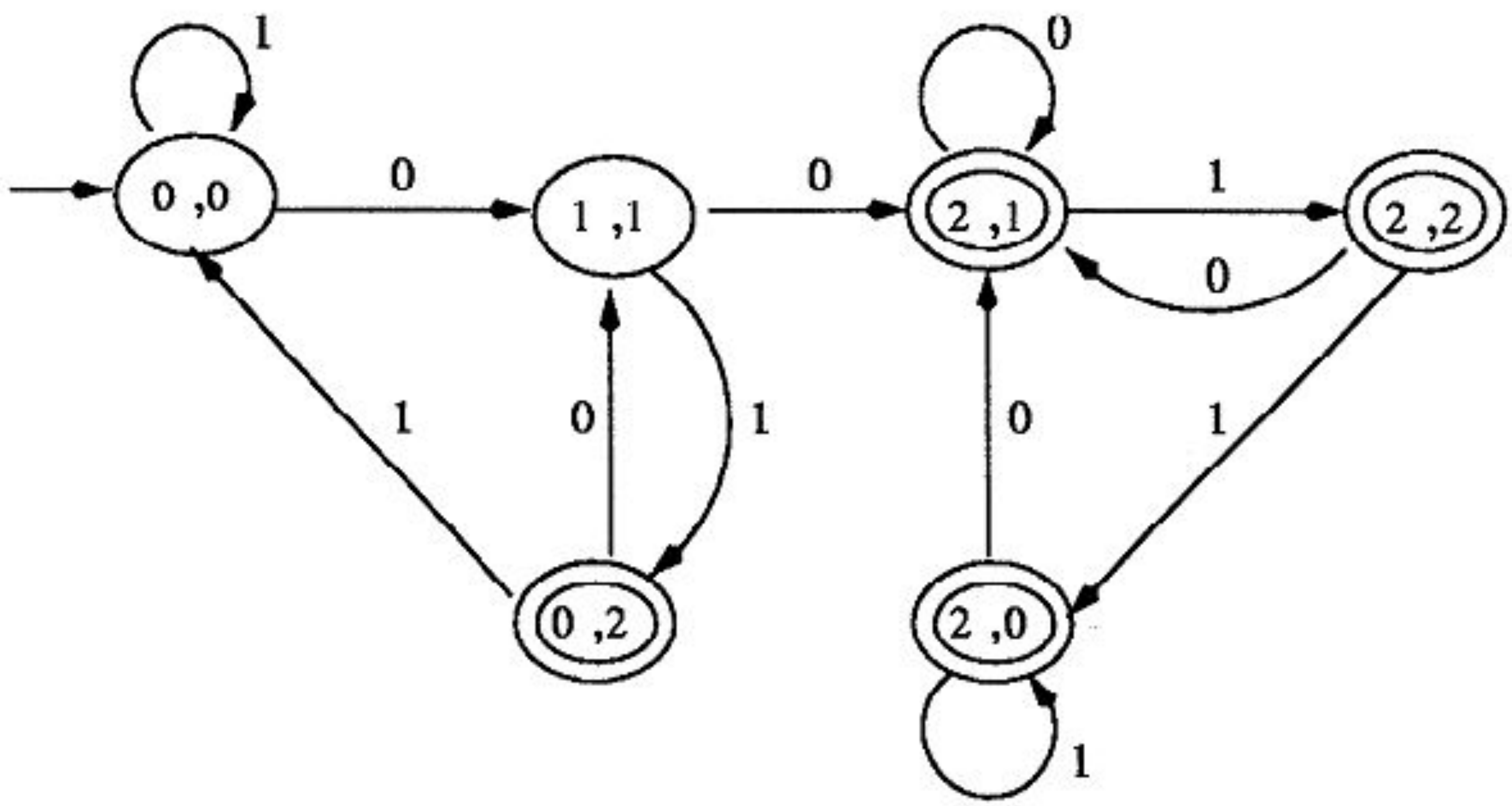
Solution of example 1

- DFA to accept $(0+1)^*01$.



Solution of example 1

- DFA to the set of all binary strings having a substring 00 or ending with 01.



Closure properties of regular languages

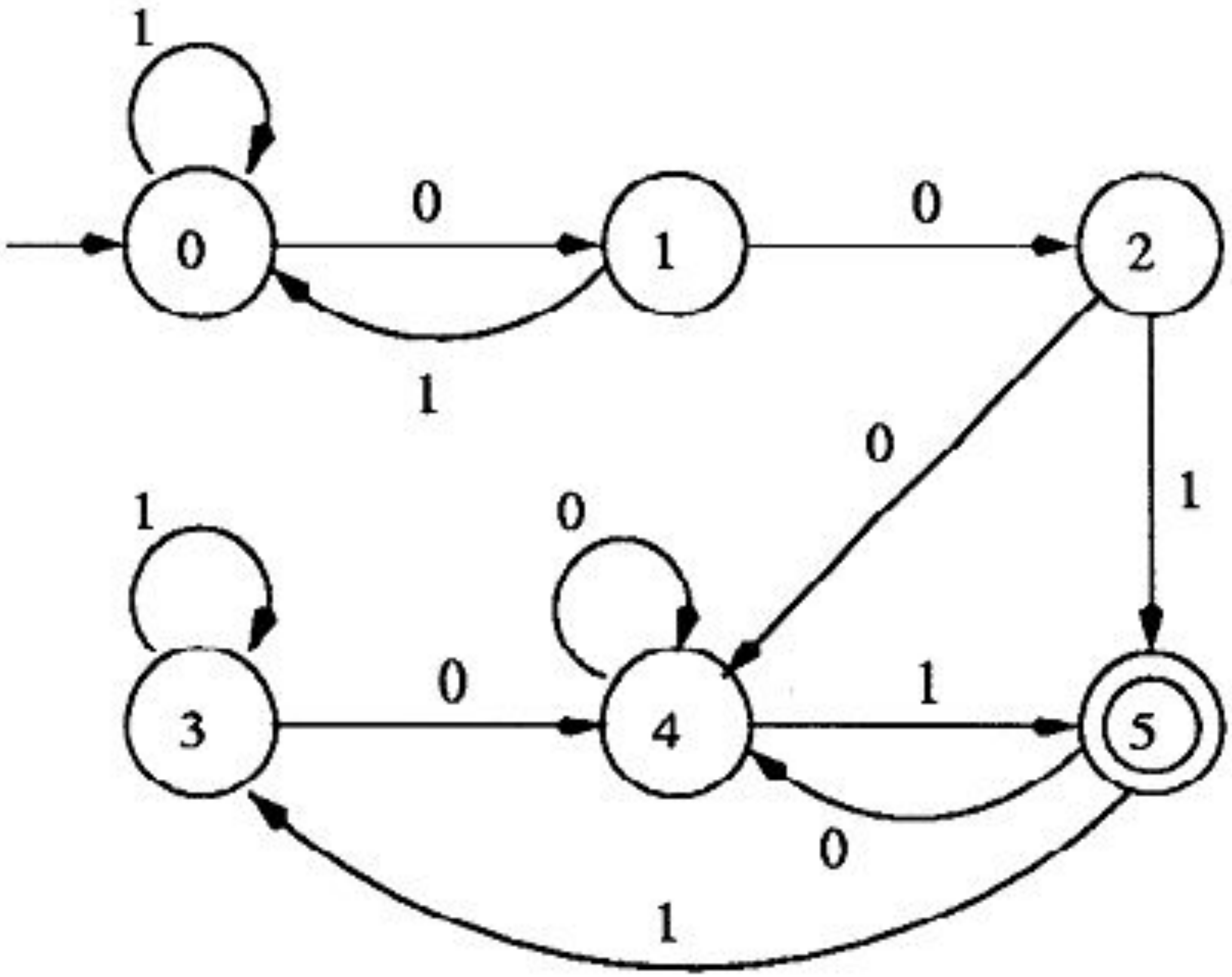
Example 2

The set of all binary strings having a substring 00 and ending with 01.

This language is the intersection of two languges $(0+1)^*00(0+1)^*$ and $(0+1)^*01$.

The transition diagram of the resulting DFA is just like that of example 1, except that the final set consists of only one state (q_2, q_2).

Another solution of example 2.



Closure properties of regular languages

Example 3

The set of all binary strings having a substring 00 but not ending with 01.

This language is the difference of language $(0+1)^*00(0+1)^*$ minus language $(0+1)^*01$.

The transition diagram of the resulting DFA is just like that of example 1, except that the final set consists of two states (q_2, q_0) and (q_2, q_1).

Closure properties of regular languages

Example 4

The set L of all binary strings in which every block of four consecutive symbols contains a substring 01.

Solution.

The condition "every block of four consecutive symbols contains a substring 01" is a global condition, which appears difficult to verify.

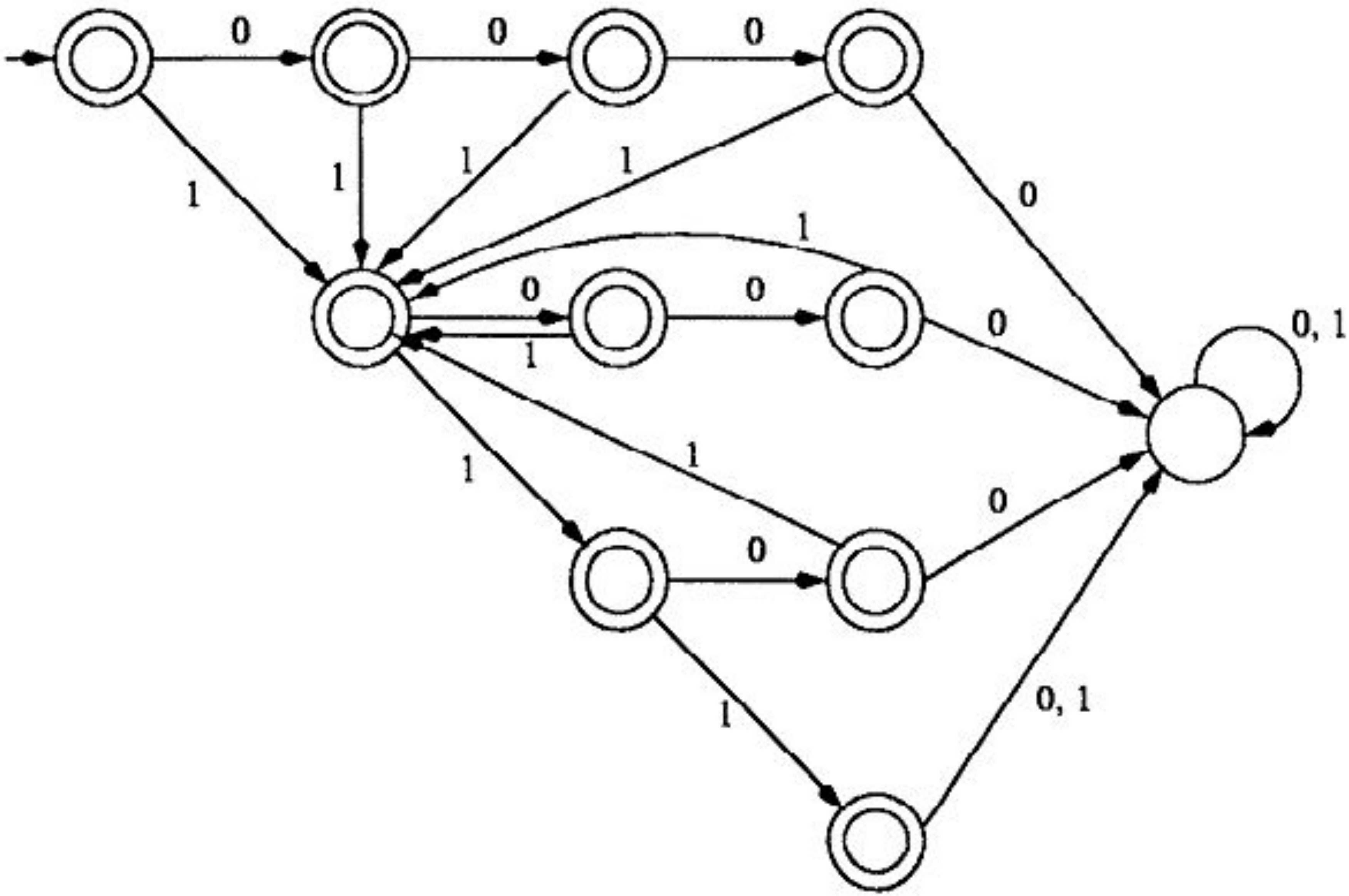
By considering the complement $\bar{L}$, we turn this condition into a simpler local condition: $\bar{L}$ contains binary strings with a substring $0000, 1000, 1100, 1110$ or 1111.

Solution of example 4

- We first construct a DFA accepting $\bar{L}$ and then change all final states into nonfinal states and all nonfinal states into final states.

A solution is shown in the following figure.

Solution of example 4



Pumping lemmas

- Not all languages are regular.

We introduce necessary condition for regularity of languages which can be used to prove that a language is non regular.

In the following, we write, for any string v^* to denote the set $\{v^*\}$.

Pumping lemmas

<u>Pumping lemma</u>. If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

<u>Proof.</u> Consider the transition diagram of M.

Since $x \in L$, the computation path π of x starts from the initial state q_0 and ends at a final state q_f.

The concatenation of the labels over the path π is exactly the string x.

? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

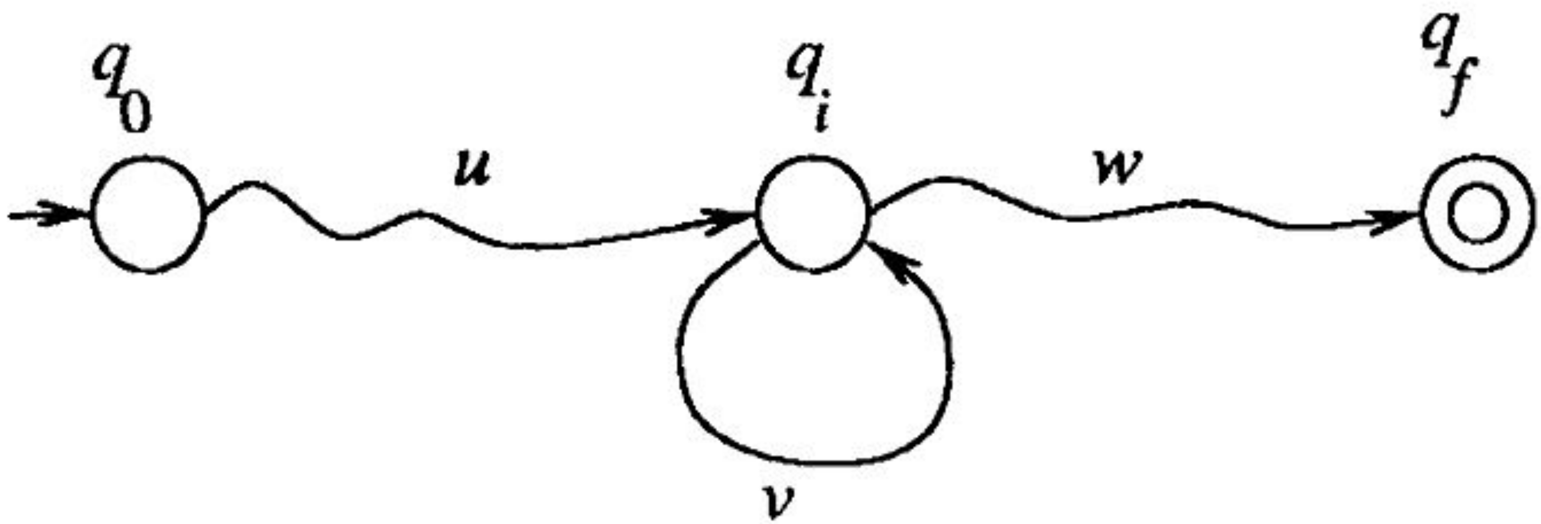
- The path π has exactly $|x|$ edges because each edge is labeled by a symbol.

Thus, the path π contains a vertex sequence of $|x| + 1$ elements.

Since $|x| \geq s$, some state q_i occurs more than once in the sequence.

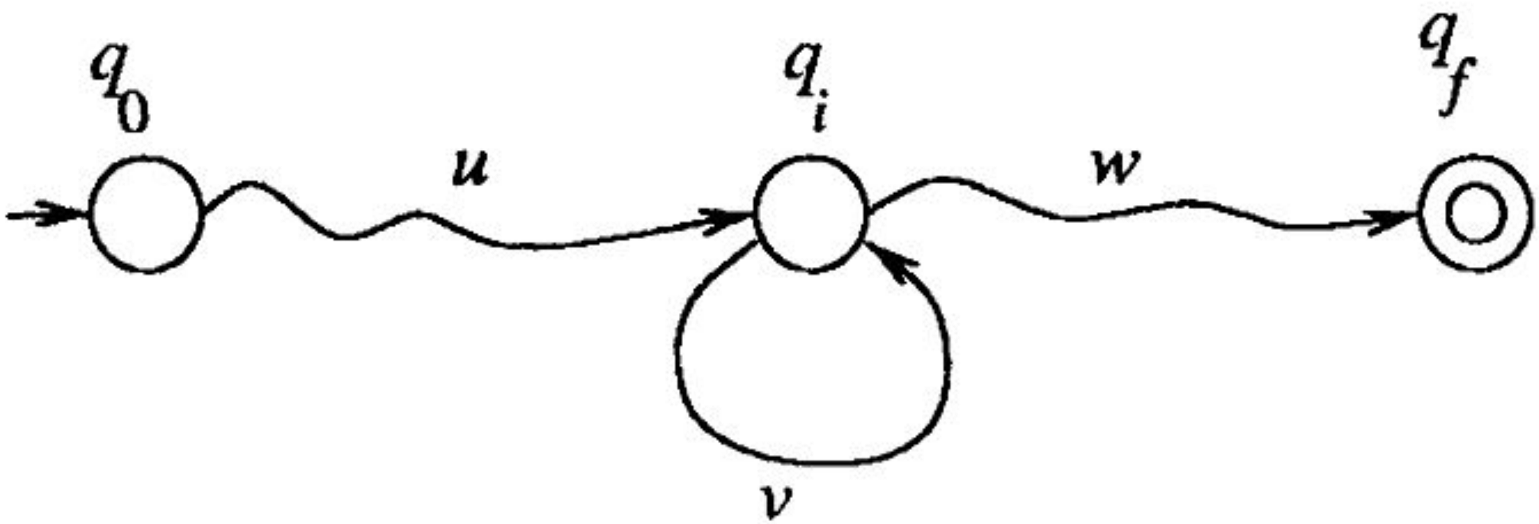
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- Since $|x| \geq s$, some state q_i occurs more than once in the sequence.



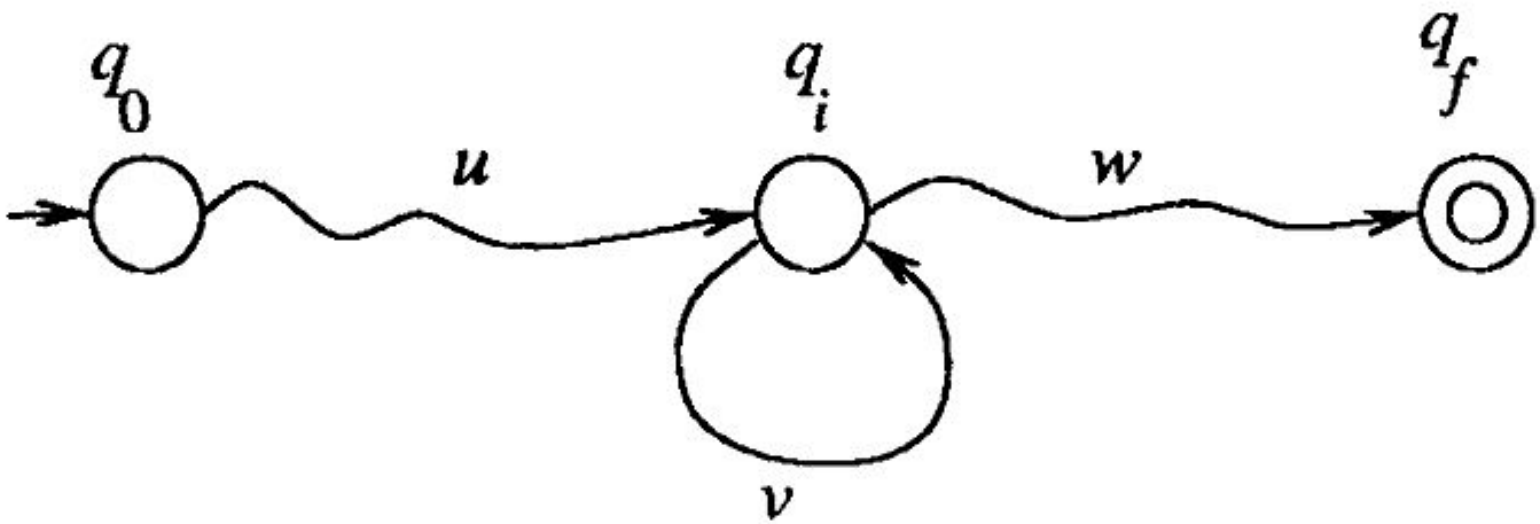
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- Break the path π into three subpaths at the first and second occurrences of q_i.



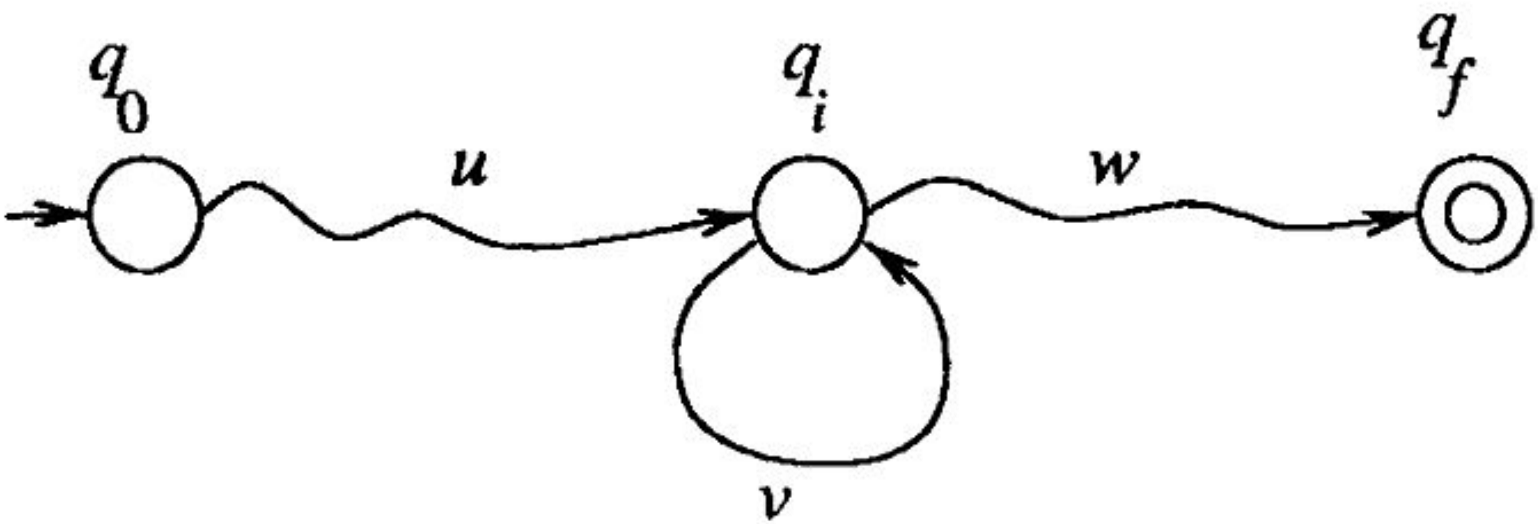
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- That is, the first subpath is from state q_0 to the first occurrence of q_i.



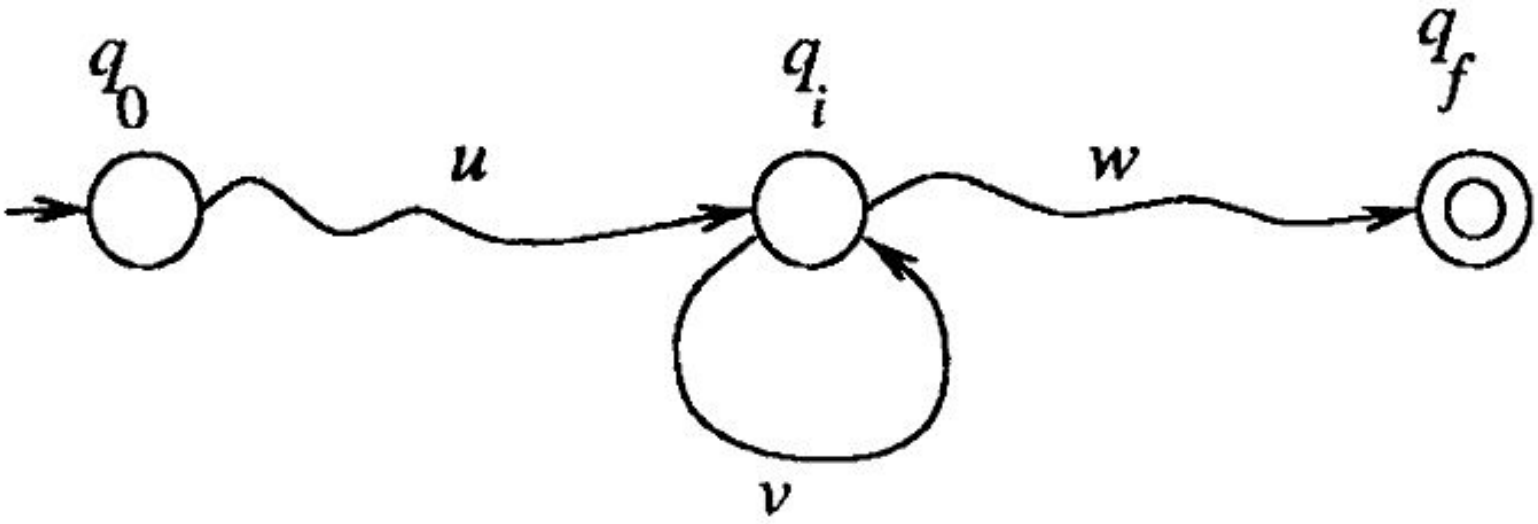
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- The second subpath is a cycle from the first q_i to the second q_i.



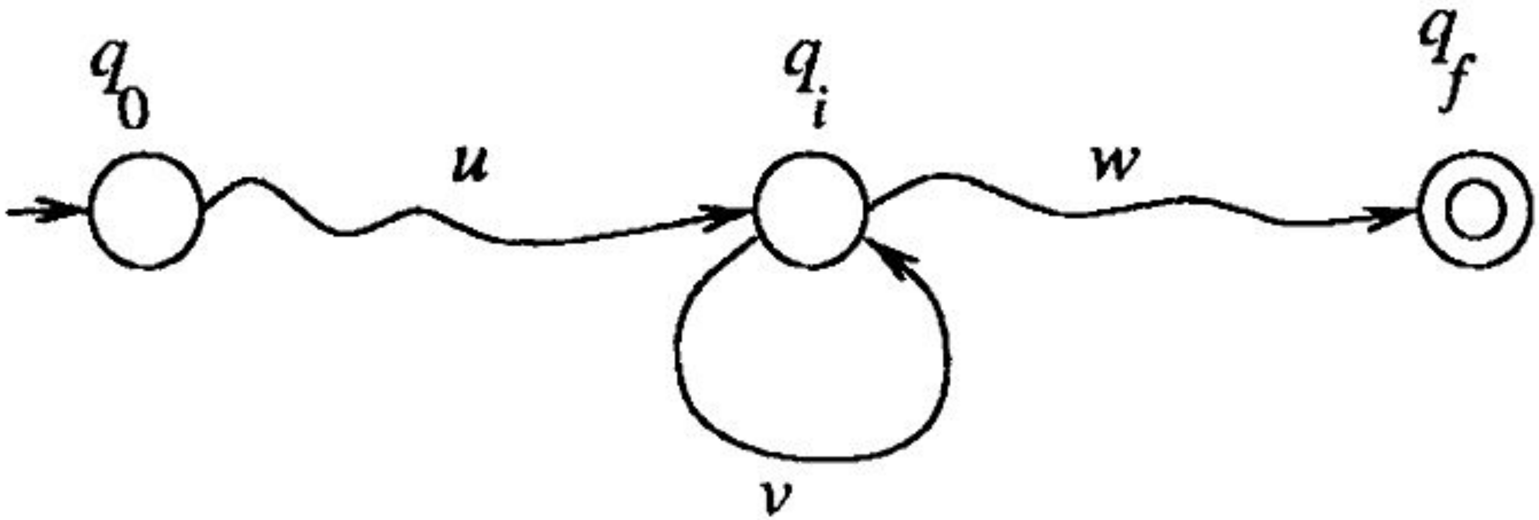
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- The third subpath is from the second q_i to q_f.



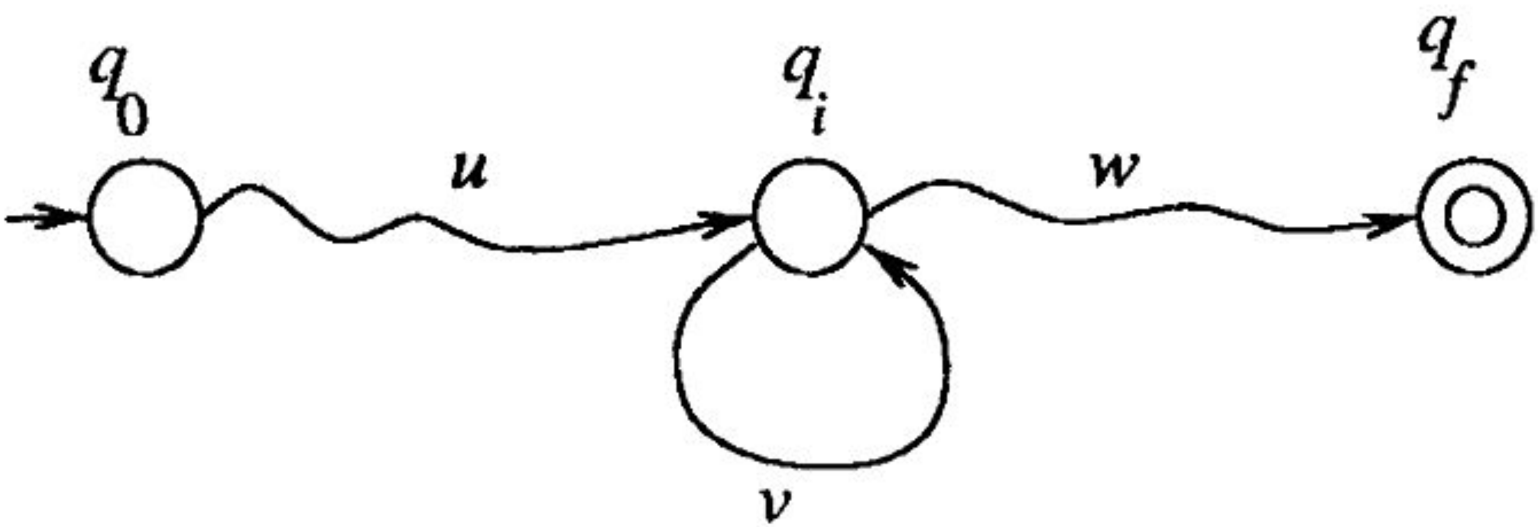
? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- Let u, v and w be the concatenations of the labels of the three subpaths, respectively. Then, $x = uvw$ and $v \neq \varepsilon$.



? If a language L is accepted by a DFA M with s states, then every string x in L with $|x| \geq s$, can be written as: $x = uvw$ such that $v \neq \varepsilon$ and $uv^*w \subseteq L$.

- Since v is associated with a cycle, we also have $uv^*w \subseteq L$. (E.g., $uv^2w \in L$ because $\delta(q_0, uv^2w) = = \delta(q_i, vvw) = \delta(q_i, vw) = \delta(q_i, w) = q_f$.) ■



Pumping lemmas

- Now, for a given language L, if we can prove that the necessary condition of the pumping lemma does not hold with respect to any $s > 0$, then L is not regular.

Pumping lemmas

- <u>Example 1</u>

$L = \{0^p \mid p -$ is a prime$\}$ is not a regular language.