LANGUAGE IF IT IS ACCEPTED BY SOME LATTICE ORDERED FINITE AUTOMATON

P.VIJAYA VANI, POKALA BHASKARUDU

Abstract : In this paper we present some interesting results relating Regular Sets Boolean Algebras and Generalized Boolean Algebras. It is trivial that power set of any set is a Boolean Algebra under set inclusion. Here we try to recognize a particular finer class of sets of Σ* and show that it is a Boolean Algebra. We consider the class of Regular sets Ł over a fixed Σ and define a relation " ≤ " on Ł by L1 ≤ L2 if L1 Í L2 . Then (Ł, ≤) is a partially ordered set. With the same order, the poset (Ł , ≤) is a lattice in which L1 Ú L2 = L1 È L2 and L1 Ù L2 = L1 Ç L2 .

Keywords: Regular Sets, Generalized Boolean Algebras.

Definition: A Non-Deterministic Finite Automaton (N D F A) is a quintuple
$M = (Q, \sum, \delta, q_0 , F)$ where
Q is a non-empty finite set called the set of states of M.
$\sum$ is a finite set called the input alphabet of M.
$\delta : Q \times \sum \rightarrow 2^Q$ is a function called the transition function of M.
q_0 is a fixed element of Q called the initial state of M
F is a subset of Q called the set of final states of M. Members of F are called accepting states.

Definition: A Deterministic Finite Automaton (D F A) is a quintuple $M = (Q, \sum, \delta, q_0 , F)$ where
Q is a non-empty finite set called the set of states of M.
$\sum$ is a finite set called the input alphabet of M.
$\delta : Q \times \sum \rightarrow Q$ is a function called the transition function of M.
q_0 is a fixed element of Q called the initial state of M
F is a subset of Q called the set of final states of M.,members of F are called accepting states.

Note: In a NDFA $M = (Q, \sum, \delta, q_0 , F)$ if $|\delta (q , a)| = 1$ for every $(q , a) \in Q \times \sum$, then M is called a deterministic finite automaton (DFA).

Definition: A Finite Automaton (FA) is either a NDFA or DFA. Formally a finite automaton (FA) consists of a finite set of states and a set of transitions from state to state that occur on input symbols chosen from an alphabet Σ. One state, usually denoted q_0, is the initial state, in which the automaton starts. Some states are designated as final or accepting states.
A Finite Automaton (FA) may be interpreted as a finite control which is in some state of Q, reading a sequence of symbols from $\sum$ written on a tape. In one move the Finite Automaton in state ' q ' and scanning symbol ' a ' enters next state $\delta (q , a)$ and moves its head one symbol to the right.

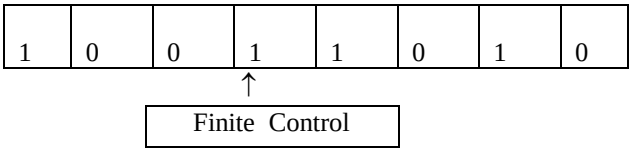


Figure – 2.1.12

Definition: A directed graph called a Transition diagram is associated with an FA as follows. The vertices of the graph correspond to the states of the FA. If there is a transition from state q to state p on input a then there is an arc labeled a from state q to state p in the transition diagram. The FA accepts a string x if the sequence of transitions corresponding to the symbols of x leads from the start state to an accepting state. Accepting states are double circled.

Example of Automaton transition:
For $M = (Q, \sum, \delta, q_0 , F)$ where
$Q = \{ q_0 , q_1 , q_2, q_3 , q_4 \}$;
$\Sigma = \{ 0 ,1 \}$;
q_0 is the start state;
$F = \{ q_2, q_4 \}$

δ as follows

δ	0	1
q_0	$\{q_0,q_3\}$	$\{q_0,q_1\}$
q_1	Ø	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_4\}$	Ø
q_4	$\{q_4\}$	$\{q_4\}$

The transition diagram is

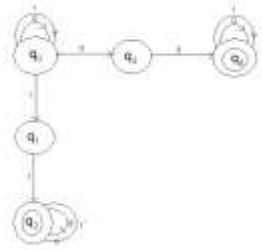


Figure- 2.1.14

Definition: A Finite Automaton (FA) accepts a string ' s ' if the sequence of transitions corresponding to the symbols of ' s ' starting from the initial state and ends at an accepting state.

Definition: A string 's' is said to be accepted by a Finite Automaton $M = (Q, \sum, \delta, q_0, F)$ if $\delta (q , s)$ contains an element of F.

Definition: Extend δ to δ^* from $Q \times \sum^* \rightarrow 2^Q$, defined as

1. $\delta^*(q , sa) = \delta (\delta^*(q , s) , a)$ for all strings s and input symbols a .
2. $\delta^*(q , \varepsilon) = q$.

Definition: The language accepted by a Finite Automaton $M = (Q, \sum, \delta, q_0 , F)$ denoted by L(M) is the set
$L (M) = \{ s / \delta^* (q_0 , s)$ contains an element of F }

Definition: A language L is said to be a regular set if it is the set accepted by some Finite Automaton.

Theorem: [Theorem 2.1 of [1]] Let L be a set accepted by a Non-Deterministic Finite Automaton. Then there exists a Deterministic Finite Automaton that accepts L
Proof: Let M = (Q, ∑, δ, q_0 , F) be a N D F A accepting L.
Define a DFA, $M^1 = (Q^1, \sum, \delta^1, q_0^1, F)$ as follows:
$F^1 = \{ A \subseteq Q / A \cap F \neq \emptyset \}$
An element of Q^1 will be denoted by [q_1 , q_2 , ….., q_i], where q_1 , q_2 , ……., q_i are in Q.
Let $q_0^1 = [q_0]$.
We define
$\delta^1 ([q_1 , q_2 , \ldots\ldots., q_i], a) = [p_1, p_2 ,\ldots\ldots\ldots, p_j]$ if and only if
$\delta(\{q_1 , q_2 , \ldots\ldots., q_i \}, a) = \{ p_1, p_2 ,\ldots\ldots\ldots, p_j \}$
that is δ^1 is applied to an element
[q_1 , q_2 , ……., q_i], of Q^1 is computed by applying δ to each state of Q represented by [q_1 , q_2 , ……., q_i].
On applying δ to each q_1 , q_2 , ……., q_i and taking the union , we get some new set of states , $p_1, p_2 ,\ldots\ldots\ldots, p_j$.
This new set has a representative $[p_1, p_2 ,\ldots\ldots\ldots, p_j]$ in Q^1 , and that element is the value of $\delta^1 ([q_1, q_2 ,\ldots, q_i], a)$
It is easy to show by induction on the length of the input string ‘s’ that $\delta^1 (q_0^1, s) = [q_1 , q_2 , ., q_i]$ if and only if $\delta (q_0 , s) = \{ q_1 , q_2 , ., q_i \}$
Basis: The result is trivial for $| s | = 0$, since $q_0^1 = [q_0]$ and s must be ε.
Induction : Suppose that the hypothesis is true for inputs of length m or less.
Let sa be a string of length m+1 with a in ∑ then
$\delta^1 (q_0^1, sa) = \delta^1 (\delta^1(q_0^1, s), a)$
By the inductive hypothesis
$\delta^1 (q_0^1, s) = [p_1, p_2 ,\ldots\ldots\ldots, p_j]$ if and only if
$\delta (q_0 , s) = \{p_1, p_2 ,\ldots\ldots\ldots, p_j \}$.
But by definition of δ^1,
$\delta^1 ([p_1, p_2 ,\ldots\ldots\ldots, p_j], a) = [r_1, r_2 ,\ldots\ldots., r_k]$ if and only if
$\delta(\{p_1, p_2 ,\ldots\ldots\ldots, p_j \}, a) = \{ r_1, r_2 ,\ldots\ldots\ldots, r_k\}$, which establishes the inductive hypothesis. Now we have only to add that $\delta^1 (q_0^1, s)$ is in F exactly when $\delta (q_0, s)$ contains a state of Q that is in F.
Thus $L (M) = L (M^1)$
Since deterministic and nondeterministic finite automaton accept the same sets, we shall not distinguish between them unless it becomes necessary, but shall simply refer to both as finite automata.

Definition: A **N D F A** with ε - transitions is a quintuple $M = (Q, \sum, \delta, q_0 , F)$ where
Q is a non-empty finite set of states
∑ is a finite input alphabet
δ is a transition function $Q \times \sum \cup \{ \varepsilon \}$ to 2^Q that is
$\delta : Q \times \sum \cup \{\varepsilon\} \rightarrow 2^Q$
q_0 is an initial state , $q_0 \in Q$
F is a subset of Q called the set of final states of M.
Members of F are called accepting states.

Definition: Let $M = (Q, \sum, \delta, q_0 , F)$ be an N D F A. Let q Є Q, ε- Closure(q) is defined as the set of all vertices p such that there is a path from q to p labeled ε.

Definition: Let M = (Q, ∑, δ, q_0 , F) be an N D F A .
Let $P \subseteq Q$.
ε-closure(P) is defined as $\cup_{p \text{ in } P}$ ε- Closure(p). Now we define δ^ as follows.

1. $\hat{\delta}$ (q , ε) = ε- Closure(q).
2. For w in Σ* and a in Σ, $\hat{\delta}$ (q , wa) = ε- Closure(P),

where P = { p / for some r in $\hat{\delta}$ (q, w), p is in δ(r , a)},
It is convenient to extend δ and $\hat{\delta}$ two sets of states by
$\delta (R, a) = \cup_{q \text{ in } R} \delta (q , a)$ and
$\hat{\delta}$ (R, w) = $\cup_{q \text{ in } R}$ $\hat{\delta}$ (q , w) for sets of states R
We define L (M) , the language accepted by
M = (Q, ∑, δ, q_0 , F) to Be
{ w / $\hat{\delta}$ (q_0,w) contains a state in F}.

2.2.5 Theorem: If L is accepted by an N D F A with ε - transitions then L is accepted by an N D F A without ε – transitions.
Proof. Let M = (Q, ∑, δ, q_0 , F) be an NDFA with ε – transitions.
Construct $M^1 = (Q, \sum, \delta^1, q_0 , F^1)$ where

$$F^1 = \begin{cases} F_1 = F \cup \{ q_0\} \text{ if } \varepsilon\text{- Closure}(q0) \text{ contains a state of F} \\ F \quad \text{otherwise} \end{cases}$$

and $\delta^1 (q, a)$ is $\hat{\delta}$ (q , a) for q in Q and a in Σ. Note that M^1 has no ε– transitions. Thus we may use δ^1 for $\delta^{1\wedge}$.
We show by induction on | x | that $\delta^1 (q_0, x) = \hat{\delta} (q_0 , x)$. However this statement may not be true for x = ε,
since $\delta^1 (q_0, \varepsilon) = \{q_0\}$, while $\hat{\delta}$ $(q_0 , x) =$ ε- closure(q_0).
We therefore begin our induction at 1.
Basis : | x | = 1. Then x is a symbol ‘a’ and
$\delta^1 (q_0, a) = \hat{\delta}$ (q_0 , a) by definition of δ^1 .
Induction : | x |> 1. Let x = wa for symbol a in Σ.
Then $\delta^1 (q_0, wa) = \delta^1 (\delta^1 (q_0, w), a)$.
By the inductive hypothesis $\delta^1 (q_0, w) = \hat{\delta}$ (q_0 , w).
Let $\hat{\delta} (q_0 , x) = P$.
We must show that $\delta^1 (P, a) = \hat{\delta} (q_0 , wa)$.

But $\delta^1(P, a) = \cup_{q \text{ in } P} \delta^1(q, a) = \cup_{q \text{ in } P} \hat{\delta}(q, a)$.
Then as
$P = \hat{\delta}(q_0, w)$ we have $\cup_{q \text{ in } P} \hat{\delta}(q, a) = \hat{\delta}(q_0, wa)$.
Thus $\delta^1(q_0, wa) = \hat{\delta}(q_0, wa)$.
For completeness we show that $\delta^1(q_0, x)$ contains a state of F^1 if and only if $\hat{\delta}(q_0, x)$ contains a state in F.
If $x = \varepsilon$, this statement is immediate from the definition of F^1. That is $\delta^1(q_0, \varepsilon) = \{q_0\}$, and q_0 is placed in F^1 whenever $\hat{\delta}(q_0, \varepsilon)$ which is ε- Closure(q_0), contains a state possibly q_0 in F. If $x \neq \varepsilon$ then x = wa for some symbol a . If $\hat{\delta}(q_0, x)$ contains a state in F, then surely $\delta^1(q_0, x)$ contains a state in F^1 . Conversely if $\delta^1(q_0, x)$ contains a state in F^1 other than q_0, then $\hat{\delta}(q_0, x)$ contains a state in F. If $\delta^1(q_0, x)$ contains q_0 and q_0 is not in F, then as $\hat{\delta}(q_0, x) = \varepsilon$- Closure($\delta(\hat{\delta}(q_0, w), a)$), the state in ε- Closure(q_0) and in F must be in $\hat{\delta}(q_0, x)$.
Thus L accepted by an N D F A with ε - transitions is accepted by an N D F A without ε – transitions.

Definition: Let ∑ be a finite set of symbols and let languages L, L_1, L_2 be sets of strings from $\sum^*$

- The concatenation of L_1 and L_2 , denoted by L_1L_2 is the set { s_1s_2 / s_1 is in L_1 and s_2 is in L_2 }.
- The Kleene Closure of L, denoted by L^* , is the set $L^* = \cup_{i=0}^{\infty} L^i$
- The Positive Closure of L, denoted by L^+ , is the set $L^+ = \cup_{i=0}^{\infty} L^i$
- $L^* = L\,L^{i-1}$ and $L^0 = \{\varepsilon\}$.

Definition : If r is a regular expression of the language L then we write L = L(r). The regular expressions are said to be equal i.e., r = s if L(r) = L (s).

Definition: Let ∑ be an alphabet, the regular expressions over ∑ are defined recursively as follows.

- ⌀ is a regular expression and denotes the empty set.
- ε is a regular expression and denotes the set { ε }.
- For each ' a ' in ∑ , ' a ' is a regular expression and denotes the set {a}.
- If l_1 and l_2 are regular expressions corresponding to the languages L_1 and L_2 respectively. Then $(l_1 + l_2)$, (l_1l_2) , l_1^* and l_2^* are regular expressions that corresponding to the sets $L_1 \cup L_2$, L_1L_2 , L_1^* and L_2^* respectively.

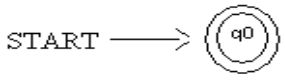
Definition : If r is a regular expression denoting the language L then we write L = L(r). We say that regular expression r and s are equal, and write r =s if L(r) = L (s).

Result: For regular expressions r , s , t corresponding to the languages L (r), L (s), L (t) respectively. It is easy to verify the following.

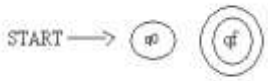
- r + s = s + r
- (r + s) + t = r +(s + t)
- (rs) t = r (st)
- r (s + t) = rs + rt
- (r + s) t = rt + st
- $\phi^* = \varepsilon$
- $(r^*)^* = r^*$
- $(\varepsilon + r)^* = r^*$
- $(r^* s^*)^* = (r + s)^*$

Theorem: Let r be a regular expression of a language L. Then there exists an N D F A with ε - transitions that accepts L (r).
Proof: We shall prove this theorem by induction on the number of operators in the regular expression ' r 'of L, that there is an N D F A with ε - transitions, having one final state and no transitions out of this final state, such that L(M) = L(r).
Suppose the expression r has zero operators.
Since r has zero operators, the expression r must be ε, ϕ, or a for ' a ' in ∑. The N D F A 's in the following figures clearly satisfy the conditions.

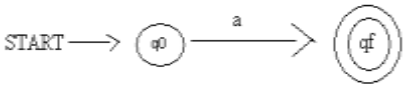
1. r = ε



2. r = ϕ



3. r = **a**



Assume that the theorem is true for regular expressions r with fewer than n operators, n > 1. Let r have n operators. There are three cases depending on the form of r.
Case 1 : Let $r = r_1 + r_2$, where r_1 and r_2 are regular expressions less than n operators. Thus there are N D F A 's $M_1 = (Q_1, \sum_1, \delta, q_1, \{f_1\})$ and
$M_2 = (Q_2, \sum_2, \delta, q_2, \{f_2\})$ with
$L(M_1) = L(r_1)$ and $L(M_2) = L(r_2)$.
Since we may rename states of a N D F A at will we may assume Q_1 and Q_2 are disjoint. Let q_0 be a new initial state and f_0 a new final state.
We construct
$M = (Q_1 \cup Q_2 \cup \{q_0, f_0\}, \sum_1 \cup \sum_2, \delta, q_0, \{f_0\})$,
Where δ is defined by

1. $\delta(q_0, \varepsilon) = \{q_1, q_2\}$.
2. $\delta(q, a) = \delta_1(q, a)$ for q in $Q_1 - \{f_1\}$ and a in $\sum_1 \cup \{\varepsilon\}$
3. $\delta(q, a) = \delta_2(q, a)$ for q in $Q_2 - \{f_2\}$ and a in $\sum_2 \cup \{\varepsilon\}$
4. $\delta(f_1, \varepsilon) = \delta_1(f_2, \varepsilon) = \{f_0\}$

We recall by the inductive hypothesis that there are no transitions out of f_1 or f_2 in M_1 or M_2 .Thus all the moves of M_1 and M_2 are present in M.

The construction of M is depicted in the following figure.

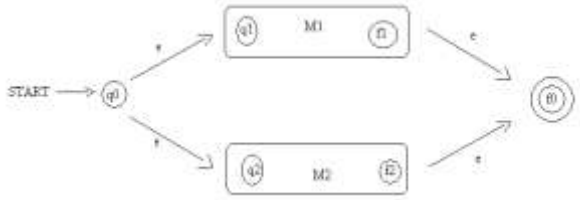


Figure 2.3.6

Thus any path in the transition diagram of M from q_0 to f_0 must begin by going to either q_1 or q_2 or ε. If the path goes to q_1, it may follow any path in M_1 to f_1 and then go to f_0 on ε. Similarly paths that begin by going to q_2 may follow any path in M_2 to f_2 and then go to f_0 on ε. These are the only paths from q_0 to f_0 . It follows immediately that there is a path labeled ' s ' in M_1 from q_1 to f_1 or a path in M_2 from q_2 to f_2 .

Hence $L(M) = L(M_1) \cup L(M_2)$ as desired.

Case 2 : $r = r_1 r_2$

Let M_1 and M_2 be as in case 1 and construct

$M = (Q_1 \cup Q_2, \Sigma_1 \cup \Sigma_2, \delta, q_1, \{f_2\})$, where

1. $\delta(q, a) = \delta_1(q, a)$ for q in $Q_1 - \{f_1\}$ and a in $\Sigma_1 \cup \{\varepsilon\}$
2. $\delta(f_1, \varepsilon) = \{q_2\}$
3. $\delta(q, a) = \delta_2(q, a)$ for q in Q_2 and a in $\Sigma_2 \cup \{\varepsilon\}$

The construction of M is given in the following figure.

Figure 2.3.6

Every path in M from q_1 to f_2 is a path labeled by some string ' s ' from q_1 to f_1, followed by the edge from f_1 to q_2 labeled ε,followed by a path labeled by some string from q_2 to f_2 Thus $L(M) = \{ xy / x \text{ is in } L(M_1) \text{ and } y \text{ is in } L(M_2) \}$ and $L(M) = L(M_1) L(M_2)$ as desired.

Case 3 : $r = r_1^*$

Let $M_1 = (Q_1 \cup \{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\})$,Where δ is given by

1. $\delta(q_0, \varepsilon) = \delta(f_1, \varepsilon) = \{q_1, f_0\}$
2. $\delta(q, a) = \delta_1(q, a)$ for q in $Q_1 - \{f\}$ and a in $\Sigma_1 \cup \{\varepsilon\}$.

The construction of M is depicted in the following figure.

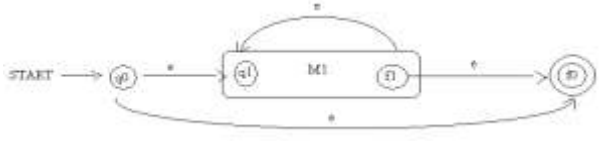


Figure 2.3.6

So any path from q_0 to f_0 consists either of a path from q_0 to f_0 on ε or a path from q_0 to q_1 on ε, followed by some number (possibly zero) of paths from q_1 to f_1 , then back to q_1 on ε, each labeled by a string in $L(M_1)$, followed by a path from q_1 to f_1 on a string in $L(M_1)$, then to f_0 on ε . Thus there is a path in M from q_0 to f_0 labeled x if and only if we can write $s = s_1 s_2 \ldots\ldots s_j$ for some $j \geq 0$ (the case j = 0 means $s = \varepsilon$) such that each s_i is in $L(M_1)$

Hence $L(M) = L(M_1)^*$ as desired.

Theorem: If L is accepted by a DFA then L is denoted by a regular expression.

Proof: Let L be the language accepted by the DFA $M = (Q, \Sigma, \delta, q_0, F)$ where $Q = \{q_1, q_2, q_3, \ldots q_n\}$. Let R^k_{ij} denote the set of all strings x such that $\delta(q_i, x) = q_j$ and if $\delta(q_i, y) = q_l$, for any y that is a prefix of x other than x or ε, then $l \leq k$. That is R^k_{ij} is the set of all strings that take the finite automaton from state q_i to state q_j with out going through any state numbered higher than k . Since there is no state numbered greater than n ,R^k_{ij} denotes all strings that take q_i to q_j .

We can define R^k_{ij} recursively,

$R^k_{ij} = R^{k-1}_{ik} (R^{k-1}_{kk})^* R^{k-1}_{kj} \cup R^{k-1}_{ij}$

$R^0_{ij} = \{a / (\delta(q_i, a) = q_j\}$ if $i \neq j$

$= \{a / (\delta(q_i, a) = q_j\} \cup \{\varepsilon\}$ if $i = j$

We show that for each i , j , k there exists a regular expression r^k_{ij} denoting the language R^k_{ij}. We proceed by induction on k.

Basis: k = 0. R^0_{ij} is a finite set of strings each of which either ε or a single symbol. Thus r^0_{ij} can be written as $a_1 + a_2 + a_3 + \ldots + a_p$ or $(a_1 + a_2 + a_3 + \ldots + a_p + \varepsilon)$ if i = j , where $\{a_1, a_2, a_3, \ldots a_p\}$ is the set of all symbols a such that $\delta(q_i, a) = q_j$.

If there are no such a's then Ø (ε or in the case i = j) serves as r^0_{ij}.

Induction: The recursive formula for R^k_{ij} given clearly involves only the regular expression operators: union, concatenation and closure. By the induction hypothesis for each l and m there exists a regular expression r^{k-1}_{lm} such that $L(r^{k-1}_{lm}) = R^{k-1}_{lm}$. Thus for r^k_{ij} we may select the regular expression

$r^k_{ij} = (r^{k-1}_{ik})(r^{k-1}_{kk})^*(r^{k-1}_{kj}) + (r^{k-1}_{ij})$, which completes the induction.

Now we observe that $L(M) = \cup_{q \text{ in } F} R^n_{1j}$ since R^n_{1j} denotes the labels of all paths from q_1 to q_j. Thus L (M) is denoted by the regular expression.

$r^n_{1j1} + r^n_{1j2} + r^n_{1j3} + \ldots\ldots + r^n_{1jp}$. where $F = \{q_{j1} + q_{j2} + q_{j3} + \ldots\ldots + q_{1jp}\}$.

Thus L accepted by a DFA is denoted by a regular expression.

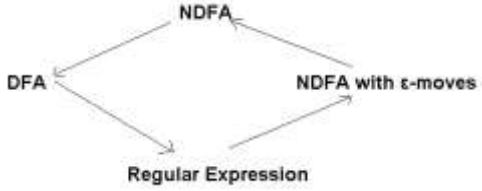


Figure 2.3.7

Thus the languages accepted by finite automata are precisely the languages denoted by regular expressions are precisely the languages denoted by regular expressions. This equivalence was the motivation for calling finite automaton languages regular languages.

Pumping Lemma is a powerful tool for proving that certain languages are non regular.

Lemma:[Lemma 3.1 of [1]] Let L be a regular language. Then there is a constant n such that if z is any word in L and$| z | \geq n$, we write z = uvw in such a way that $|uv| \leq n$, $| v | \geq 1$, and for all $i \geq 0$, uv^iw is in L. Furthermore, n is no greater than the number of states of the smallest FA accepting L.

Example: The language $L = \{ a^{j^2} / j$ is an integer, $j \geq 1\}$, which consists of all strings of a 's whose length is a perfect square, is not regular.

Solution: Assume L is a regular language and let m be the integer in the pumping lemma,

let $z = a^m$. By the pumping lemma , a^m may be written as uvw, where $1 \leq | v | \leq m$

and uv^iw is in L for all j. In particular, let j = 2.

However, $m^2 < | uv^2w | \leq m^2 + m < (m+1)^2$.

That is, the length of uv^2w lies properly between m^2 and $(m+1)^2$ and is thus not a perfect square.

Thus uv^2w is not in L , a contradiction .

We conclude that L is not regular.

Theorem: The regular languages are closed under union, concatenation and kleene closure.

Proof: Claim 1 : Union of Regular languages is regular.

Let L_1 be L (M_1) for DFA $M_1 = (Q_1, \Sigma_1, \delta, q_1 , \{f_1\})$ and

Let L_2 be L (M_2) for DFA $M_2 = (Q_2, \Sigma_2, \delta, q_2 , \{f_2\})$.

If r_1 and r_2 are regular expressions denoting regular languages L_1 and L_2 then $r_1 + r_2$ denotes $L_1 \cup L_2$ = L (M) for DFA

$M = (Q_1 \cup Q_2 \cup \{q_0 , f_0 \}, \Sigma_1 \cup \Sigma_2 , \delta , q_0 , \{f_0\})$, where δ is

1. $\delta (q_0 , \varepsilon) = \{ q_1 , q_2 \}$.
2. $\delta (q , a) = \delta_1 (q , a)$ for q in $Q_1 - \{ f_1 \}$ and a in $\Sigma_1 \cup \{\varepsilon\}$
3. $\delta (q , a) = \delta_2 (q , a)$ for q in $Q_2 - \{ f_2 \}$ and a in $\Sigma_2 \cup \{\varepsilon\}$
4. $\delta (f_1 , \varepsilon) = \delta_1 (f_2 , \varepsilon) = \{ f_0 \}$

So $L_1 \cup L_2$ is also regular. Thus Union of Regular languages is regular.

Hence regular languages are closed under union.

Claim 2 : Concatenation of Regular languages is regular.

Let L_1 be L (M_1) for DFA $M_1 = (Q_1, \Sigma_1, \delta, q_1 , \{f_1\})$ and

Let L_2 be L (M_2) for DFA $M_2 = (Q_2, \Sigma_2, \delta, q_2 , \{f_2\})$.

If r_1 and r_2 are regular expressions denoting regular languages L_1 and L_2 then r_1r_2 denotes L_1L_2 = L (M) for DFA

$M = (Q_1 \cup Q_2 , \Sigma_1 \cup \Sigma_2 , \delta , q_1 , \{f_2\})$, where δ is

1. $\delta (q , a) = \delta_1 (q , a)$ for q in $Q_1 - \{ f_1 \}$ and a in $\Sigma_1 \cup \{\varepsilon\}$
2. $\delta (f_1 , \varepsilon) = \{ q_2\}$
3. $\delta (q , a) = \delta_2 (q , a)$ for q in Q_2 and a in $\Sigma_2 \cup \{\varepsilon\}$

So L_1L_2 is also regular. Thus Concatenation of Regular languages is regular.

Hence regular languages are closed under Concatenation.

Claim 3 : Kleene Closure of Regular languages is regular.

Let L_1 be L (M_1) for DFA $M_1 = (Q_1, \Sigma_1, \delta, q_1 , \{f_1\})$

If r_1 is a regular expression denoting regular language L_1 then r_1^*denotes $L_1^* = L = L (M)$ for DFA

$M = (Q_1 \cup \{q_0 , f_0 \}, \Sigma_1 , \delta , q_0 , \{f_0\})$

Where δ is given by

1. $\delta (q_0 , \varepsilon) = \delta (f_1 , \varepsilon) = \{q_1 , f_0 \}$
2. $\delta (q, a) = \delta_1(q , a)$ for q in $Q_1 - \{ f \}$ and a in $\Sigma_1 \cup \{ \varepsilon \}$.

So $L = L_1^*$ is also regular. Thus Kleene Closure of Regular languages is regular.

Hence regular languages are closed under Kleene Closure.

Theorem: The class of regular languages is closed under complementation that is if L is a regular language and $L \subseteq \Sigma^*$ then $\Sigma^* - L$ is a regular language.

Proof : Let L be L (M) for DFA $M =(Q, \Sigma_1, \delta, q_0 , F)$ and let $L \subseteq \Sigma^*$ To prove the class of regular languages is closed under complementation, we construct a DFA $M^1 = (Q, \Sigma, \delta, q_0 , Q - F)$.

First we may assume $\Sigma_1 = \Sigma$, for if there are symbols in Σ_1 not in Σ, we may delete all transitions of M on symbols not in Σ. The fact that $L \subseteq \Sigma^*$ assures us that we shall not there by change the language of M.

If there are symbols in Σ not in Σ_1, then none of these symbols appear in words of L. We may therefore introduce a " dead state " ' d ' into M with $\delta (d , a) = d$ for all a in Σ and $\delta (q , a) = d$ for all q in Q and a in $\Sigma - \Sigma_1$. Now to accept $\Sigma^* - L$, we complement the final state of M that is $M^1 = (Q, \Sigma, \delta, q_0 , Q - F)$ then M^1 accepts a word w if and only if $\delta (q_0 , w)$ is in $Q - F$ that is w is in $\Sigma^* - L$. Thus if L is a regular language and $L \subseteq \Sigma^*$ then $\Sigma^* - L$ is a regular language that is the class of regular languages is closed under complementation.

Theorem: The class of regular languages are closed under intersection.

Proof: Let L_1 be L (M_1) for DFA $M_1 = (Q_1, \Sigma, \delta, q_1 ,F_1)$ and

Let L_2 be L (M_2) for DFA $M_2 = (Q_2, \Sigma, \delta, q_2 , F_2)$.

Consider $L_1 \cap L_2$

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2 ,$$

where the over bar denotes the complementation with respect to an alphabet including the alphabets L_1 and L_2 . Thus Closure under intersection follows from closure under union and complementation.

Definition Let **Ł** be the class of regular languages. Define a relation " $\leq$ " on **Ł** by $L_1 \leq L_2$ if $L_1 \subseteq L_2$.

Proposition: If **Ł** be the class of regular languages then (**Ł**, $\leq$) is a partially ordered language.

Proof: For any $L_1 \in$ **Ł** , we have by the definition of " is contained in "

$L_1 \subseteq L_2 \Rightarrow L_1 \leq L_2$

Thus ' $\leq$ ' is reflexive.
Suppose $L_1 \leq L_2$ and $L_2 \leq L_1$
$\Rightarrow L_1 \subseteq L_2$ and $L_2 \subseteq L_1$
Now by the definition of equality in languages
Since $L_1 \subseteq L_2$ and $L_2 \subseteq L_1$ imply $L_1 = L_2$
Thus ' $\leq$ ' is antisymmetric.
Suppose $L_1 \leq L_2$ and $L_2 \leq L_3 \Rightarrow L_1 \subseteq L_2$ and $L_2 \subseteq L_3$
Now $L_1 \subseteq L_2 \subseteq L_3 \Rightarrow L_1 \subseteq L_3$
$\Rightarrow L_1 \leq L_3$
Thus ' $\leq$ ' is transitive.
Therefore (**Ł**, $\leq$) is a partially ordered language.

Theorem: Let **Ł** be the class of regular languages. Then the poset **Ł** is a lattice in which $L_1 \vee L_2 = L_1 \cup L_2$ and $L_1 \wedge L_2 = L_1 \cap L_2$.
Proof: Now we show that **Ł** is a lattice .
Claim 1: l. u. b. $\{ L_1 , L_2 \} = L_1 \vee L_2 = L_1 \cup L_2$
Let $L = L_1 \cup L_2$
By definition of union on languages
$L_1 \subseteq L$ and $L_2 \subseteq L$
$\Rightarrow L_1 \leq L$ and $L_2 \leq L$
Suppose there exists L_* such that $L_1 \leq L_*$ and $L_2 \leq L_*$
$\Rightarrow L_1 \subseteq L_*$ and $L_2 \subseteq L_*$
$\Rightarrow L_1 \cup L_* = L_*$ and $L_2 \cup L_* = L_*$
Consider $L_* = L_1 \cup L_*$
$= L_1 \cup (L_2 \cup L_*)$
$= (L_1 \cup L_2) \cup L_*$
$\Rightarrow L_1 \cup L_2 \subseteq L_*$
$\Rightarrow L \subseteq L_*$
$\Rightarrow L \leq L_*$
Thus l. u. b. $\{ L_1 , L_2 \} = L_1 \vee L_2 = L_1 \cup L_2 = L$
Claim 2: g. l. b. $\{ L_1 , L_2 \} = L_1 \wedge L_2 = L_1 \cap L_2$
Let $L = L_1 \cap L_2$
By definition of intersection on languages
$L \subseteq L_1$ and $L \subseteq L_2$
$\Rightarrow L \leq L_1$ and $L \leq L_2$
Suppose there exists L_* such that $L_* \leq L_1$ and $L_* \leq L_2$
$\Rightarrow L_* \subseteq L_1$ and $L_* \subseteq L_2$
$\Rightarrow L_* = L_* \cap L_1$ and $L_* = L_* \cup L_2$
Consider $L_* = L_* \cap L_2$
$= (L_* \cap L_1) \cap L_2$
$= L_* \cap (L_1 \cap L_2)$
$\Rightarrow L_* \subseteq L_1 \cap L_2$
$\Rightarrow L_* \subseteq L \Rightarrow L_* \leq L$
Thus g. l. b. $\{ L_1 , L_2 \} = L_1 \wedge L_2 = L_1 \cap L_2 = L$
Hence Ł is a lattice in which $L_1 \vee L_2 = L_1 \cup L_2$ and $L_1 \wedge L_2 = L_1 \cap L_2$.

Note: For **Ł**, the class of regular languages over a fixed Σ , $\emptyset \in$ **Ł** is the minimal element of **Ł** as $\emptyset \leq$ **Ł** i.e. $\emptyset \subseteq$ **Ł** for all $L \in$ **Ł** and $\Sigma^* \in$ **Ł** is the maximal element of **Ł** as $L \leq \Sigma^*$ for all $L \in$ **Ł.**

Definition: The lattice **Ł** of regular languages is said to be a lattice monoid if it satisfies the following conditions.
(i) $L * \phi = \phi * L = \phi \forall L \in$ **Ł.**
(ii) $L_1 * (L_2 \vee L_3) = (L_1 * L_2) \vee (L_1 * L_3)$ and
(iii).$(L_2 \vee L_3) * L_1 = (L_2 * L_1) \vee (L_3 * L_1)$
$\forall L_1 , L_2 , L_3 \in$ **Ł**
and $*$ is a binary operation
(in particular on **Ł** ,$*$ is concatenation).

Note : Σ^* is a free monoid generated by Σ with the concatenation operation.

Definition: A Lattice Ordered Finite Automaton (LA) is a quintuple $M = (Q, \Sigma, \delta, I , F)$ where
Q is a non-empty finite set called the set of states of M.
Σ is a finite set called the input alphabet of M.
$\delta : Q \times \Sigma \times Q \rightarrow [\phi, \Sigma^*]$ is a function called the fuzzy transition function of M.
I: $Q \rightarrow [\phi, \Sigma^*]$ is a fuzzy subset of Q called the fuzzy initial state set of M.
F: $Q \rightarrow [\phi, \Sigma^*]$ is a fuzzy subset of Q called the fuzzy final state set of M.

Definition : Extend δ to δ^* from $Q \times \Sigma \times Q \rightarrow [\phi, \Sigma^*]$, defined as
1 .$\delta^*(q , \varepsilon, p) = \Sigma^*$ if $q = p$
$= \phi$ otherwise
2 .$\delta^*(q, sa, p) = \vee [\delta^*(q, s , r) \wedge \delta^*(r, a, p)]$, $s \in \Sigma^*$, $a \in \Sigma$, $p, q, r \in Q$.

Note: For any two strings $s, w \in \Sigma^*$ and for all $p, q \in Q$, $\delta^*(q, sw, p) = \vee[\delta^*(q, s , r) \wedge \delta^*(r, a, p)]$, $r \in Q$.

Definition: The language accepted by a lattice ordered finite automaton (LA) $M = (Q, \Sigma, \delta, I , F)$ denoted by L(M) is the set $L(M) = \{s/I(q) * \delta^*(q, s , p) * F(p)\}$ and is called ℓ-language.

Definition: A ℓ-language L is said to be ℓ- regular language if it is accepted by some lattice ordered finite automaton (LA) i.e. L = L(M).

Note: Two lattice ordered finite automaton LA's M_1 and M_2 are said to be equivalent if they accept same ℓ-language i.e. $L(M_1) = L(M_2)$.

REFERENCES

1. Adaricheva, K.V., and J.B.Nation. "Reflections on lower bounded lattices", AlgebraUniversalis 53(2005).307-330.
2. Beauquier.D. and Pin. J.E . "Factors of words".Volume 372 of Lecture notes in computer Science, pages 63-79, Berlin,1989. SpringVerlag.

3. Bernstein E,Vazrani, "Quantum complexitytheory". SIAMJ comp,1997.
4. Campeanu.C. and Ho. Nu.W.H.,Culik.K. "Statecomplexity of basic operations on finitelanguages". In O. Boldt and H.Jurgensen,Proceedings of the 4th Internationalworkshop on implementing. Automata,number 2214 in LNCS,
5. Campeanu.C. and Ho. W.H. "The maximumstate complexity for finite language" .Journalof Automata, Languages andCombinatorics,9(2-3) 189-202,september2004.
6. Castro JL, Delgado M, Mantas C T. "A newapproach for the execution and adjustmentof a fuzzy algorithm" Fuzzy set system 2001.
7. Champarnadh.J.M. and Pin. J.E ."A maximumproblem on finite automata". Discreteapplied mathematics.
8. Crespi areghizzi.S., Pradella. M.,Pietro. P ." Associative definitions of programming languages". Computer languages, 26:105-123, 2000.
9. Eilenberg.S. "Automata, Languages, and machines", Vol. A. Acd Press,1974.
10. Feiner.l. "Hierarchies of Boolean algebras".11. Freese,R.,Jezek.J. amd Nation.J.B."Free Lattices", Mathematical Surveys and Monographs, vol. 42, Amer.Math.soc.,Providence, RI, 1995.
12. Gile C,Omlin C,Thornber K K. "Equivalence in knowledge representation: Automata, Recurrent neural network and dynamical fuzzy systems."
13. Gratzer,G.,"General Lattice Theory", SecEdn,Birkhauser Verlag, Basel, 1998
14. Gunter Pilz."Near-rings"
15. Hanf.W. "Model-theoretic methods in the study of elementary logic", Theory of models (Proc.1963 internet. Sympos., Berkeley), North-Holland,Amsteerdam,1965,pp.132 145.MR 35#1457.
16. Holcombe."Near-rings associated withautomata", San B163-166.
17. Holzer.M. and M.Kutrib. "State complexity of basic operations on nondeterministic finite automata"
18. Hopcroft J.E,Ullman "Introduction to Automata theory. Language and Computation."
19. Hopcroft.J.E. and Ullman.J.D. "Introduction to automata theory, Languages, and computation",addision-Weslwy,1979.
20. Joshi.A.K., Levy.L.S., and Yueh.K. "Local constraints in the syntax and semantics of programming languages".
21. Lee E T,Zadeh L.A "A note on fuzzy languages" inf Sci, 1969.
22. Li.Y M,Pedrycz W.F. "Fuzzy finite automata and fuzzy regular expressions with membership values in lattice -ordered monoids."
23. Li.Y.M., Z.K.Shi, "Finite Automata with values in lattice-monoid and their languages."

P.Vijaya Vani
Faculty, Dept of IBS
Acharya Nagarajuna University , Guntur,
Andhra Pradesh, India.

Pokala Bhaskarudu
Lecturer in Mathematics
S V Arts College (T T D),Tirupati