Sinhgad College of Engineering Department of Computer Engineering

Name of Student: **Prashant Kumar**

Roll No: **405B049**

PRN Number: **71813716H**

Seat No: **B150234347**

Class: **BE** Div.: **2** Batch: **B**

Subject: **Soft Computing and Optimization Algorithms**

Name of Laboratory: **Laboratory Practice IV**

List of Assignments

S.NO.	Title of Assignment	Remark	Signature
1	Implementation of Fuzzy Operations: Union, Intersection, Complement and Difference operations on fuzzy sets.		
2	Implementation of Fuzzy Relation: by Cartesian product of any two fuzzy sets and perform max-min composition on fuzzy relations.		
3	Perform Functions of Logic Gates using McCulloh-Pitts Neural Network (model)		
4	Write a program for solving linearly separable problem using Perceptron Model.		
5	Study and Implement optimization problem using Partial Swarm Optimization		

Experiment No. 1

1. **Aim:** Implementation of Fuzzy Operations.

2. **Objectives:**

   - Provide an understanding of the basic mathematical elements of fuzzy sets.
   - Understand and analyse concepts of fuzzy set.
   - To use fuzzy set operations to implement current computing techniques used in fuzzy computing.

3. **Outcomes:** The students will be able to

   - Learn mathematical basis as well as the general principles of various soft computing techniques.
   - To analyze the applications using fuzzy set which uses current techniques, skills, and tools necessary for computing.
   - To identity and solves the engineering problems using the fuzzy set theory and identify the differences and similarities between fuzzy sets and classical sets theories.

4. **Hardware/Software Required :**JAVA/ C/C++/MATLAB/**OCTAVE**

5. **Theory:**

Fuzzy Logic:

Fuzzy logic is an organized method for dealing with imprecise data. It is a multivalued logic that allows intermediate values to be defined between conventional solutions.
In classical set theory, the membership of elements in a set is assessed in binary terms according to a bivalent condition — an element either belongs or does not belong to the set. By contrast, fuzzy set theory permits the gradual assessment of the membership of elements in a set; this is described with the aid of a membership function valued in the real unit interval [0, 1].

Bivalent Set Theory can be somewhat limiting if we wish to describe a 'humanistic' problem mathematically. For example, Fig 1 below illustrates bivalent sets to characterize the temperature of a room. The most obvious limiting feature of bivalent sets that can be seen clearly from the diagram is that they are mutually exclusive - it is not possible to have membership of more than one set. Clearly, it is not accurate to define a transition from a quantity such as 'warm' to 'hot' by the application of one degree Fahrenheit of heat. In the real world a smooth (unnoticeable) drift from warm to hot would occur.

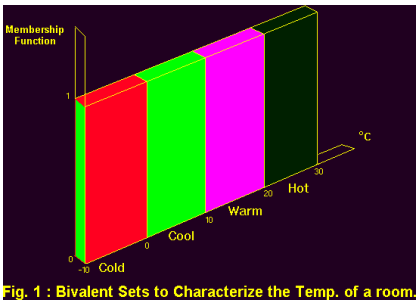


Fig. 1 : Bivalent Sets to Characterize the Temp. of a room.

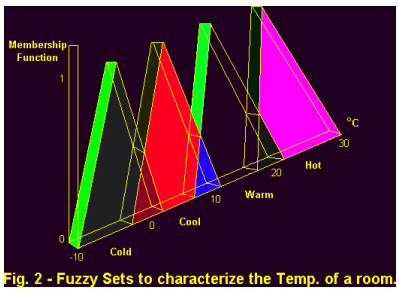


Fig. 2 - Fuzzy Sets to characterize the Temp. of a room.

Fuzzy Sets:

A fuzzy set is a pair (U, m) where U is a set and $m: U \to [0,1]$.

For each $x \in U$ the value $m(x)$ is called the degree of membership of x in (U, m). For a finite set $U = \{x_1, \ldots, x_n\}$ the fuzzy set (U, m) is often denoted by $\{m(x_1)/x_1, \ldots, m(x_n)/x_n\}$.

Let $x \in U$. Then,

- x is called not included in the fuzzy set (U, m) if $m(x) = 0$
- x is called fully included if $m(x) = 1$
- x is called a fuzzy member $0 < m(x) < 1$.

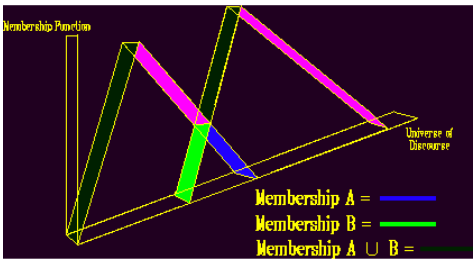
Example: $A = \{(x_1, 0.1), (x_2, 0.7), (x_3, 1), (x_4, 0)\}$

$B = \{(x_1, 0.4), (x_2, 0.3), (x_3, 1), (x_4, 0.2)\}$

1. **Union:**

Union of two fuzzy sets A and B is denoted as $A \cup B$ and is defined as,

$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$

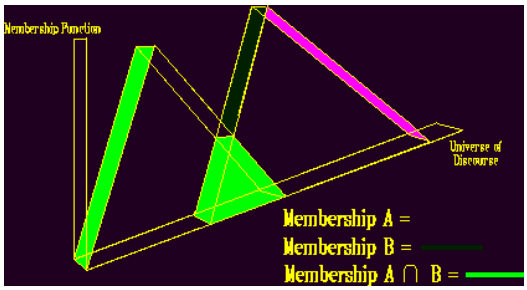


Example: $\tilde{A} \cup \tilde{B} = \{(x_1, 0.4), (x_2, 0.7), (x_3, 1), (x_4, 0.2)\}$

2. Intersection

Union of two fuzzy sets A and B is denoted $\tilde{A} \cap \tilde{B}$ and is defined as,

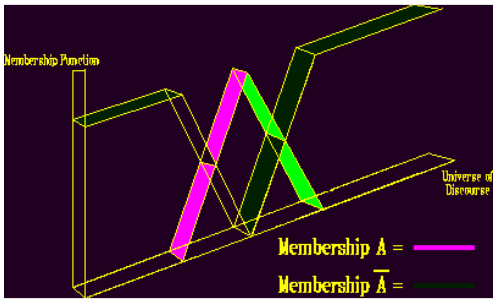
$$\mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)]$$



Example: $\tilde{A} \cap \tilde{B} = \{(x_1, 0.1), (x_2, 0.3), (x_3, 1), (x_4, 0)\}$

3. Complement

Complement of a fuzzy A denoted $\bar{A}$ and is defined as,



Example: $\bar{A} = \{(x_1, 0.9), (x_2, 0.3), (x_3, 0), (x_4, 1)\}$

4. Algebraic Sum

Algebraic sum of two fuzzy sets A and B is denoted by $A + B$ and is defined as,

$$\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$$

Example: $A + B = \{(x_1, 0.46), (x_2, 0.79), (x_3, 1), (x_4, 0.2)\}$

5. Algebraic Product

Algebraic product of two fuzzy sets A and B denoted by $A.B$ and is defined as,

$$\mu_{A.B}(x) = \mu_A(x).\mu_B(x)$$

Example: $A.B = \{(x_1, 0.04), (x_2, 0.21), (x_3, 1), (x_4, 0)\}$

6. Conclusion

The concepts of union, intersection and complement are implemented using fuzzy sets which helped to understand the differences and similarities between fuzzy set and classical set theories. It provides the basic mathematical foundations to use the fuzzy set operations.

7. Viva Questions:

- What are the properties of Fuzzy set?
- What is De Morgan‘s Law in Crisp Set?
- What is the difference between the crisp set and fuzzy set?

CODE :

```
# Union of Two Fuzzy Sets
A = dict()
B = dict()
Y = dict()

A = {"a": 0.2, "b": 0.3, "c": 0.6, "d": 0.6}
B = {"a": 0.9, "b": 0.9, "c": 0.4, "d": 0.5}

print('The First Fuzzy Set is :', A)
print('The Second Fuzzy Set is :', B)


for A_key, B_key in zip(A, B):
    A_value = A[A_key]
    B_value = B[B_key]

    if A_value > B_value:
        Y[A_key] = A_value
    else:
        Y[B_key] = B_value

print('Fuzzy Set Union is :', Y)

# OUTPUT :
# The First Fuzzy Set is : {'a': 0.2, 'b': 0.3, 'c': 0.6, 'd': 0.6}
# The Second Fuzzy Set is : {'a': 0.9, 'b': 0.9, 'c': 0.4, 'd': 0.5}
# Fuzzy Set Union is : {'a': 0.9, 'b': 0.9, 'c': 0.6, 'd': 0.6}

# Intersection of Two Fuzzy Sets
A = dict()
B = dict()
Y = dict()

A = {"a": 0.2, "b": 0.3, "c": 0.6, "d": 0.6}
B = {"a": 0.9, "b": 0.9, "c": 0.4, "d": 0.5}

print('The First Fuzzy Set is :', A)
print('The Second Fuzzy Set is :', B)


for A_key, B_key in zip(A, B):
    A_value = A[A_key]
    B_value = B[B_key]

    if A_value < B_value:
        Y[A_key] = A_value
    else:
        Y[B_key] = B_value
print('Fuzzy Set Intersection is :', Y)

# OUTPUT :
# The First Fuzzy Set is : {'a': 0.2, 'b': 0.3, 'c': 0.6, 'd': 0.6}
# The Second Fuzzy Set is : {'a': 0.9, 'b': 0.9, 'c': 0.4, 'd': 0.5}
# Fuzzy Set Intersection is : {'a': 0.2, 'b': 0.3, 'c': 0.4, 'd': 0.5}

# Difference Between Two Fuzzy Sets
A = dict()
Y = dict()
```

```
A = {"a": 0.2, "b": 0.3, "c": 0.6, "d": 0.6}

print('The Fuzzy Set is :', A)


for A_key in A:
   Y[A_key]= 1-A[A_key]

print('Fuzzy Set Complement is :', Y)



# OUTPUT :
# The Fuzzy Set is : {'a': 0.2, 'b': 0.3, 'c': 0.6, 'd': 0.6}
# Fuzzy Set Complement is : {'a': 0.8, 'b': 0.7, 'c': 0.4, 'd': 0.4}



# Difference Between Two Fuzzy Sets
A = dict()
B = dict()
Y = dict()

A = {"a": 0.2, "b": 0.3, "c": 0.6, "d": 0.6}
B = {"a": 0.9, "b": 0.9, "c": 0.4, "d": 0.5}

print('The First Fuzzy Set is :', A)
print('The Second Fuzzy Set is :', B)


for A_key, B_key in zip(A, B):
    A_value = A[A_key]
    B_value = B[B_key]
    B_value = 1 - B_value

    if A_value < B_value:
        Y[A_key] = A_value
    else:
        Y[B_key] = B_value

print('Fuzzy Set Difference is :', Y)


# OUTPUT :
# The First Fuzzy Set is : {"a": 0.2, "b": 0.3, "c": 0.6, "d": 0.6}
# The Second Fuzzy Set is : {"a": 0.9, "b": 0.9, "c": 0.4, "d": 0.5}
# Fuzzy Set Difference is : {"a": 0.1, "b": 0.1, "c": 0.6, "d": 0.5}
```

Experiment No. 2

1. **Aim:** Implementation of fuzzy relations (Max-Min Composition)

2. **Objectives:**

- Provide an understanding of the basic mathematical elements of the theory of fuzzy sets.
- To introduce the ideas of fuzzy sets, fuzzy logic.
- To implement applications using the fuzzy set operations.

3 **Outcomes:** The will be able to

- Understand the fuzzy arithmetic concepts.
- Become familiar with fuzzy relations and the properties of these relations.
- Identify, formulate and solve engineering problem.

4 **Software Required:** JAVA/ C/C++/MATLAB/**OCTAVE**

5 **Theory:**

Max-Min Composition of fuzzy Relations

Fuzzy relation in different product space can be combined with each other by the operation called –Composition‖. There are many composition methods in use, e.g. max-product method, max-average method and max-min method. But max-min composition method is best known in fuzzy logic applications.

Definition:

Composition of Fuzzy Relations

- Consider two fuzzy relation; R (X ×Y) and Q (Y ×Z), then a relation S (X × Z), can be expressed as max-min composition

S = R o Q

$\mu S(x, z) = \max\text{-}\min [\mu R (x, y), \mu Q (y, z)]$

$= V [\mu R (x, y) \wedge \mu Q (y, z)]$

- If algebraic product is adopted, then max-product composition is adopted:

S = R o Q

$\mu S (x, z) = \max [\mu R (x, y). \mu Q (y, z)]$

$= V [\mu R (x, y). \mu Q (y, z)]$

- The max-min composition can be interpreted as indicating the strength of the existence of relation between the elements of X and Z.
- Calculations of (R o Q) are almost similar to matrix multiplication.

Crisp relation:

Crisp relation is defined on the Cartesian product of two sets. Consider,

$$X \times Y = \{(x,y) | x \in X, y \in Y\}$$

The relation on this Cartesian product will be,

$$\mu_R = \begin{cases} 1, (x,y) \in R \\ 0, (x,y) \notin R \end{cases}$$

Example: Let X={1,4,5} and Y={3,6,7} then for relation R=x<y,

$$R = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Fuzzy relation:

Let $X, Y = R$ be universal sets then,

$$R = \{((x,y), \mu_R(x,y)) \mid (x,y) \in X \times Y\}$$

Is called a fuzzy relation in $X \times Y = R$

Example: Let X= {1, 2, 3} and Y={1,2}

If $\mu_R(x,y) = e^{-(x-y)^2}$, then

$$R = \left\{ \frac{e^{-(1-1)^2}}{(1,1)}, \frac{e^{-(1-2)^2}}{(1,2)}, \frac{e^{-(2-1)^2}}{(2,1)}, \frac{e^{-(2-2)^2}}{(2,2)}, \frac{e^{-(3-1)^2}}{(3,1)}, \frac{e^{-(3-2)^2}}{(3,2)} \right\}$$

$$R = \begin{bmatrix} 1 & 0.37 \\ 0.37 & 1 \\ 0.02 & 0.37 \end{bmatrix}$$

Max-Min Composition:

Let X,Y and Z be universal sets and let R and Q be relations that relate them as,

$$R = \{(x,y) | x \in X, y \in Y, R \subset X \times Y\}$$

$$Q = \{(y,z) | y \in Y, z \in Z, Q \subset Y \times Z\}$$

Then S will be a relation that relates elements of X with elements of Z as,

$$S = R \circ Q$$

$$S = \{(x,z) | x \in X, z \in Z, S \subset X \times Z\}$$

Max min composition is then defined as,

$$\mu_S(x,z) = \max\left\{\min\left\{\mu_R(x,y), \mu_Q(y,z)\right\}\right\}$$

Example: $R = \begin{bmatrix} 0.6 & 0.5 & 0.4 \\ 0.2 & 0.1 & 0.2 \end{bmatrix}$ and $Q = \begin{bmatrix} 0.2 & 0.6 \\ 0.1 & 0.3 \\ 0.7 & 0.5 \end{bmatrix}$

$$S = \begin{bmatrix} 0.4 & 0.6 \\ 0.2 & 0.2 \end{bmatrix}$$

6. Conclusion

With the use of fuzzy logic principles max min composition of fuzzy set is calculated which describes the relationship between two or more fuzzy sets.

7. Viva Questions:

- What is the main difference between the probability and fuzzy logic?
- What are the types of fuzzy logic sets?
- Who is the founder of fuzzy logic?

```
import numpy as np

# Max-Min Composition given by Zadeh
def maxMin(x, y):
    z = []
    for x1 in x:
        for y1 in y.T:
            z.append(max(np.minimum(x1, y1)))
    return np.array(z).reshape((x.shape[0], y.shape[1]))

# Max-Product Composition given by Rosenfeld
def maxProduct(x, y):
    z = []
    for x1 in x:
        for y1 in y.T:
            z.append(max(np.multiply(x1, y1)))
    return np.array(z).reshape((x.shape[0], y.shape[1]))

# 3 arrays for the example
r1 = np.array([[1, 0, .7], [.3, .2, 0], [0, .5, 1]])
r2 = np.array([[.6, .6, 0], [0, .6, .1], [0, .1, 0]])
r3 = np.array([[1, 0, .7], [0, 1, 0], [.7, 0, 1]])

print "R1oR2 => Max-Min :\n" + str(maxMin(r1, r2)) + "\n"
print "R1oR2 => Max-Product :\n" + str(maxProduct(r1, r2)) + "\n\n"

print "R1oR3 => Max-Min :\n" + str(maxMin(r1, r3)) + "\n"
print "R1oR3 => Max-Product :\n" + str(maxProduct(r1, r3)) + "\n\n"

print "R1oR2oR3 => Max-Min :\n" + str(maxMin(r1, maxMin(r2, r3))) + "\n"
print "R1oR2oR3 => Max-Product :\n" + str(maxProduct(r1, maxProduct(r2, r3))) + "\n\n"
```

ANS

```
R1oR2 => Max-Min :
[[ 0.6  0.6  0. ]
 [ 0.3  0.3  0.1]
 [ 0.   0.5  0.1]]

R1oR2 => Max-Product :
[[ 0.6   0.6   0.  ]
 [ 0.18  0.18  0.02]
 [ 0.    0.3   0.05]]

R1oR3 => Max-Min :
[[ 1.   0.   0.7]
 [ 0.3  0.2  0.3]
 [ 0.7  0.5  1. ]]

R1oR3 => Max-Product :
[[ 1.    0.    0.7 ]
 [ 0.3   0.2   0.21]
 [ 0.7   0.5   1.  ]]

R1oR2oR3 => Max-Min :
[[ 0.6  0.6  0.6]
 [ 0.3  0.3  0.3]
 [ 0.1  0.5  0.1]]

R1oR2oR3 => Max-Product :
[[ 0.6    0.6    0.42 ]
 [ 0.18   0.18   0.126]
 [ 0.035  0.3    0.05 ]]
```

CODE :

```
import numpy as np

# Max-Min Composition given by Zadeh
def maxMin(x, y):
    z = []
    for x1 in x:
        for y1 in y.T:
            z.append(max(np.minimum(x1, y1)))
    return np.array(z).reshape((x.shape[0], y.shape[1]))

# Max-Product Composition given by Rosenfeld
def maxProduct(x, y):
    z = []
    for x1 in x:
        for y1 in y.T:
            z.append(max(np.multiply(x1, y1)))
    return np.array(z).reshape((x.shape[0], y.shape[1]))

# 3 arrays for the example
r1 = np.array([[1, 0, .7], [.3, .2, 0], [0, .5, 1]])
r2 = np.array([[.6, .6, 0], [0, .6, .1], [0, .1, 0]])
r3 = np.array([[1, 0, .7], [0, 1, 0], [.7, 0, 1]])

print "R1oR2 => Max-Min :\n" + str(maxMin(r1, r2)) + "\n"
print "R1oR2 => Max-Product :\n" + str(maxProduct(r1, r2)) + "\n\n"

print "R1oR3 => Max-Min :\n" + str(maxMin(r1, r3)) + "\n"
print "R1oR3 => Max-Product :\n" + str(maxProduct(r1, r3)) + "\n\n"

print "R1oR2oR3 => Max-Min :\n" + str(maxMin(r1, maxMin(r2, r3))) + "\n"
print "R1oR2oR3 => Max-Product :\n" + str(maxProduct(r1, maxProduct(r2, r3))) + "\n\n"
```

OUTPUT :

```
# R1oR2 => Max-Min :
# [[ 0.6  0.6  0. ]
#  [ 0.3  0.3  0.1]
#  [ 0.   0.5  0.1]]

# R1oR2 => Max-Product :
# [[ 0.6   0.6   0.  ]
#  [ 0.18  0.18  0.02]
#  [ 0.    0.3   0.05]]


# R1oR3 => Max-Min :
# [[ 1.   0.   0.7]
#  [ 0.3  0.2  0.3]
#  [ 0.7  0.5  1. ]]

# R1oR3 => Max-Product :
# [[ 1.    0.    0.7 ]
#  [ 0.3   0.2   0.21]
#  [ 0.7   0.5   1.  ]]
```

```
# R1oR2oR3 => Max-Min :
# [[ 0.6  0.6  0.6]
#  [ 0.3  0.3  0.3]
#  [ 0.1  0.5  0.1]]

# R1oR2oR3 => Max-Product :
# [[ 0.6    0.6    0.42 ]
#  [ 0.18   0.18   0.126]
#  [ 0.035  0.3    0.05 ]]
```

Experiment No. 3

1. **Aim: Logic gates** implementation using Mc-Culloch pits neural Model.

2. **Objectives:**

   The student will be able to obtain the fundamentals and different architecture
   - **To implement AND logic gate**
   - **To implement OR logic gate**
   - **To implement ANDNOT logic gate**
   - **To implement XOR logic gate**
   - The student will have a broad knowledge in developing the different algorithms for neural networks.

3. **Outcomes:** The students will be able to,

   - Describe the relation between real brains and simple artificial neural network models.
   - Understand the role of neural networks in engineering.
   - Apply the knowledge of computing and engineering concept to this discipline.

4. **Software Required:** C/C++/JAVA/ MATLAB/**OCTAVE**

5. **Theory:**

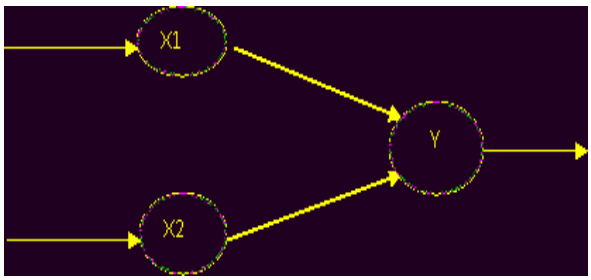
   Neural network was inspired by the design and functioning of human brain and components.

   Definition: Information processing model that is inspired by the way biological nervous system (i.e the brain) process information is called Neural Network.

   Neural Network has the ability to learn by examples. It is not designed to perform fix/specific task, rather task which need thinking (e.g. Predictions).

   ANN is composed of large number of highly interconnected processing elements (neurons) working in unison to solve problems. It mimics human brain. It is configured for special application such as pattern recognition and data classification through a learning process. ANN is 85-90% accurate.

Basic Operation of a Neural Network:



X1 and X2 – input neurons.

Y- Output neuron

Weighted interconnection links- W1 and W2.

Net input calculation is:

Yin= x1w1+x2w2

Output is:

y=f(Yin)

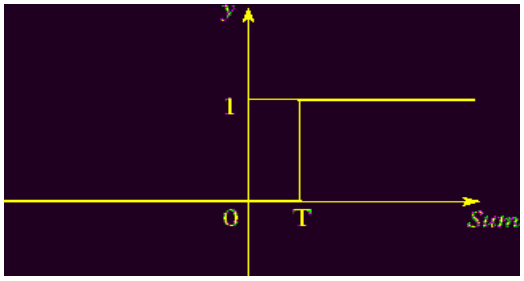
Output= function

The McCulloch-Pitts Model of Neuron:

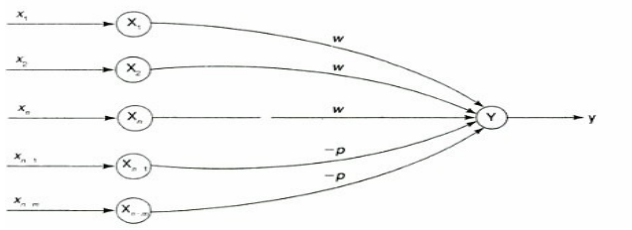
The early model of an artificial neuron is introduced by Warren McCulloch and Walter Pitts in 1943. The McCulloch-Pitts neural model is also known as linear threshold gate. It is a neuron of a set of inputs I1,I2,I3…Im and one output y . The linear threshold gate simply classifies the set of inputs into two different classes. Thus the output y is binary. Such a function can be described mathematically using these equations:

$$Sum = \sum_{i=1}^{n} I_i W_i$$

$$y = f(Sum)$$



W1,W2…Wm are weight values normalized in the range of either (0,1) or (-1,1) and associated with each input line, Sum is the weighted sum, and T is a threshold constant. The function f is a linear step function at threshold T as shown in figure



A simple M-P neuron is shown in the figure.

It is excitatory with weight (w>0) / inhibitory with weight –p (p<0).

In the Fig., inputs from x1 to xn possess excitatory weighted connection and Xn+1 to xn+m has inhibitory weighted interconnections.

Since the firing of neuron is based on threshold, activation function is defined as

$$f(y_{in}) = \begin{cases} 1 & \text{if } y_{in} \geq \theta \\ 0 & \text{if } y_{in} < \theta \end{cases}$$

For inhibition to be absolute, the threshold with the activation function should satisfy the following condition:

$$\theta > nw - p$$

Output will fire if it receives "k" or more excitatory inputs but no inhibitory inputs where

$$kw \geq \theta > (k-1)\, w$$

- The M-P neuron has no particular training algorithm.
- An analysis is performed to determine the weights and the threshold.
- It is used as a building block where any function or phenomenon is modelled based on a logic function.

Problem Statement: Implement XOR function using MP model

Truth table for XOR function is:

X1	X2	Y
0	0	0
0	1	1
1	0	1
1	1	0

Activation function Yin is as follows:

$Yin=x_1w_1+x_2w_2$

As we know,

Let $Z_1=$  and $Z_2=$ 

X1	X2		Z1
0	0		0
0	1		0
1	0		1
1	1		0

For Z1,

$W_{11}=1$ and $W_{12}=-1$ $\Theta=1$

X1	X2	Z2
0	0	0
0	1	1
1	0	0
1	1	0

For Z2,

W_{21}=-1 and W_{22}=1 Θ=1

Y=Z1+Z2

Z1	Z2		Y
0	0		0
0	1		1
1	0		1
1	1		0

For Y,

W_{11}=1 and W_{12}=1 Θ=1

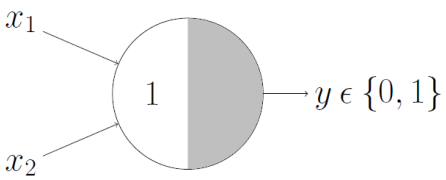
6. Conclusion:

Mc-Culloch pits Model is implemented for AND/OR/ANDNOT/ XOR function by using the thresholding activation function. Activation of M-P neurons is binary (i.e) at any time step the neuron may fire or may not fire. Threshold plays major role here.

We then established the concept of MuCulloch-Pitts neuron, the first ever mathematical model of a biological neuron. We represented a bunch of boolean functions using the M-P neuron. We also tried to get a geometric intuition of what is going on with the model, using 3D plots. In the end, we also established a motivation for a more generalized model, the one and only *artificial neuron/perceptron* model.

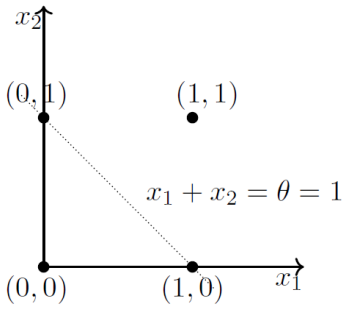
7. Viva Questions:

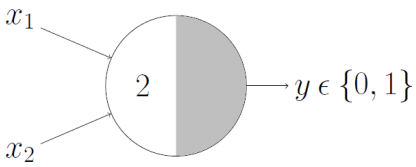
- What are Neural Networks? What are the types of neural networks?
- How are Artificial Neural Networks different from Normal Computers?
- What is simple Artificial Neuron?
- What is meant by training of artificial neural networks?



OR function

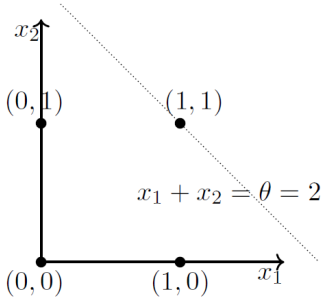
$$x_1 + x_2 = \sum_{i=1}^{2} x_i \geq 1$$





AND function

$$x_1 + x_2 = \sum_{i=1}^{2} x_i \geq 2$$



%AND function using McCulloch-Pitts neuron

```
clear;
clc;
% Getting weights and threshold value
disp('Enter the weights');
w1=input('Weight w1=');
w2=input('Weight w2=');
disp('Enter threshold value');
theta=input('theta=');
y=[0 0 0 0];
x1=[1 1 0 0];
x2=[1 0 1 0];
z=[1 0 0 0];
con=1;
while con
        zin = x1*w1+x2*w2;
        for i=1:4
                if zin(i)>=theta
                y(i)=1;
        else y(i)=0;
                end
        end
        disp('Output of net=');
        disp(y);
        if y==z
                con=0;
        else
        disp('Net is not learning Enter another set of weights and threshold value');
        w1=input('Weight w1=');
        w2=input('Weight w2=');
        theta=input('theta=');
        end
end
disp('McCulloch Pitts Net for AND function');
disp('Weights of neuron');
disp(w1);
disp(w2);
disp('Threshold value=');
disp(theta);
```

Output :-

```
Enter the weights
Weight w1=1
Weight w2=1
Enter threshold value
theta=2
Output of net= 1 0 0 0
McCulloch Pitts Net for AND function
Weights of neuron
1
1
Threshold value=
2
```

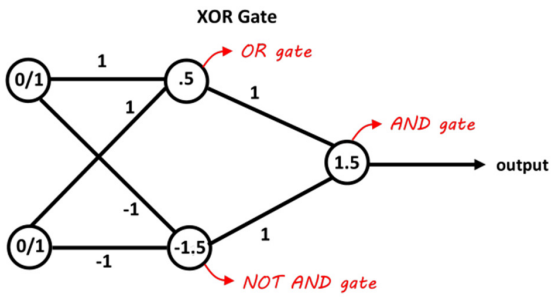
%ANDNOT function using McCulloch-Pitts neuron

```
clear;
clc;
% Getting weights and threshold value
disp('Enter the weights');
w1=input('Weight w1=');
w2=input('Weight w2=');
disp('Enter threshold value');
theta=input('theta=');
y=[0 0 0 0];
x1=[0 0 1 1];
x2=[0 1 0 1];
z=[0 0 1 0];
con=1;
while con
zin = x1*w1+x2*w2;
for i=1:4
if zin(i)>=theta
y(i)=1;
else y(i)=0;
end
end
disp('Output of net=');
disp(y);
if y==z
        con=0;
else
```

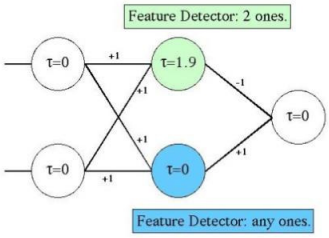
```
disp('Net is not learning Enter another set of weights and threshold value');
w1=input('Weight w1=');
w2=input('Weight w2=');
thete=input('theta=');
end
end
disp('McCulloch Pitts Net for ANDNOT function');
disp('Weights of neuron');
disp(w1);
disp(w2);
disp('Threshold value=');
disp(theta);
```

Output :-

```
Enter the weights
Weight w1=1
Weight w2=1
Enter threshold value
theta=1
Output of net= 0 1 1 1
Net is not learning Enter another set of weights and threshold value
Weight w1=1
Weight w2=-1
theta=1
Output of net=0 0 1 0
McCulloch Pitts Net for ANDNOT function
Weights of neuron
1
-1
Threshold value=
1
```



XOR Network



% XOR function using McCulloch-Pitts neuron

```
clear;
clc;
% Getting weights and threshold value
disp('Enter the weights');
w11=input('Weight w11=');
w12=input('Weight w12=');
w21=input('Weight w21=');
w22=input('Weight w22=');
v1=input('Weight v1=');
v2=input('Weight v2=');
disp('Enter threshold value');
theta=input('theta=');
x1=[0 0 1 1];
x2=[0 1 0 1];
z=[0 1 1 0];
con=1;
while con
zin1 = x1*w11+x2*w21;
zin2 = x1*w21+x2*w22;
for i=1:4
if zin1(i)>=theta
y1(i)=1;
else y1(i)=0;
end
if zin2(i)>=theta
y2(i)=1;
else y2(i)=0;
end
end
yin=y1*v1+y2*v2;
for i=1:4
if yin(i)>=theta;
y(i)=1;
else
y(i)=0;
end
end
```

```
disp('Output of net=');
disp(y);
if y==z
con=0;
else
disp('Net is not learning Enter another set of weights and threshold value');
w11=input('Weight w11=');
w12=input('Weight w12=');
w21=input('Weight w21=');
w22=input('Weight w22=');
v1=input('Weight v1=');
v2=input('Weight v2=');
theta=input('theta=');
end
end
disp('McCulloch Pitts Net for XOR function');
disp('Weights of neuron Z1');
disp(w11);
disp(w21);
disp('Weights of neuron Z2');
disp(w12);
disp(w22);
disp('Weights of neuron Y');
disp(v1);
disp(v2);
disp('Threshold value=');
disp(theta);
```

Output :-

```
Enter the weights
Weight w11=1
Weight w12=-1
Weight w21=-1
Weight w22=1
Weight v1=1
Weight v2=1
Enter threshold value
theta=1
Output of net= 0 1 1 0
McCulloch Pitts Net for XOR function
Weights of neuron z1
1
-1
Weights of neuron z2
-1
1
Weights of neuron y
1
1
Threshold value= 1
```

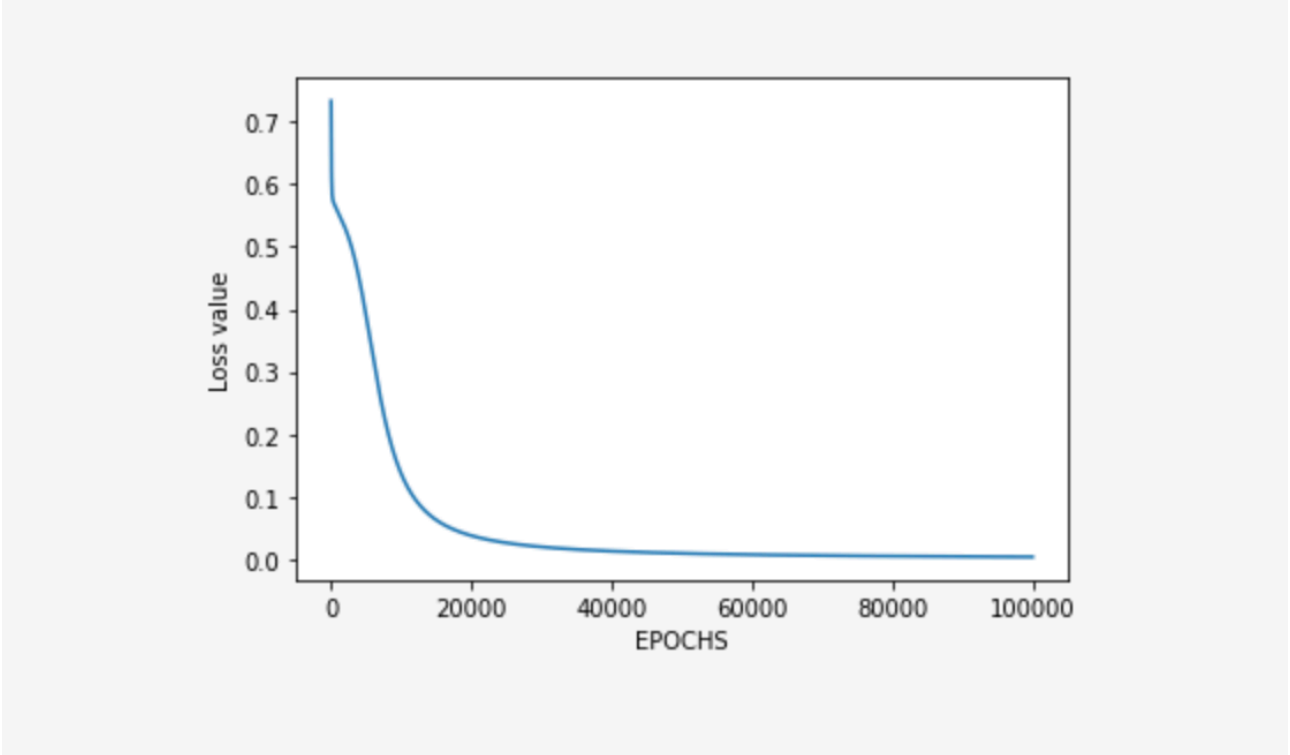
CODE :

```
#And operation

import numpy as np
from matplotlib import pyplot as plt

# Sigmoid Function
def sigmoid(z):
    return 1 / (1 + np.exp(-z))

# Initialization of the neural network parameters
# Initialized all the weights in the range of between 0 and 1
# Bias values are initialized to 0
def initializeParameters(inputFeatures, neuronsInHiddenLayers, outputFeatures):
    W1 = np.random.randn(neuronsInHiddenLayers, inputFeatures)
    W2 = np.random.randn(outputFeatures, neuronsInHiddenLayers)
    b1 = np.zeros((neuronsInHiddenLayers, 1))
    b2 = np.zeros((outputFeatures, 1))

    parameters = {"W1" : W1, "b1": b1,
                  "W2" : W2, "b2": b2}
    return parameters

# Forward Propagation
def forwardPropagation(X, Y, parameters):
    m = X.shape[1]
    W1 = parameters["W1"]
    W2 = parameters["W2"]
    b1 = parameters["b1"]
    b2 = parameters["b2"]

    Z1 = np.dot(W1, X) + b1
    A1 = sigmoid(Z1)
    Z2 = np.dot(W2, A1) + b2
    A2 = sigmoid(Z2)

    cache = (Z1, A1, W1, b1, Z2, A2, W2, b2)
    logprobs = np.multiply(np.log(A2), Y) + np.multiply(np.log(1 - A2), (1 - Y))
    cost = -np.sum(logprobs) / m
    return cost, cache, A2

# Backward Propagation
def backwardPropagation(X, Y, cache):
    m = X.shape[1]
    (Z1, A1, W1, b1, Z2, A2, W2, b2) = cache

    dZ2 = A2 - Y
    dW2 = np.dot(dZ2, A1.T) / m
    db2 = np.sum(dZ2, axis = 1, keepdims = True)

    dA1 = np.dot(W2.T, dZ2)
    dZ1 = np.multiply(dA1, A1 * (1- A1))
    dW1 = np.dot(dZ1, X.T) / m
    db1 = np.sum(dZ1, axis = 1, keepdims = True) / m

    gradients = {"dZ2": dZ2, "dW2": dW2, "db2": db2,
                 "dZ1": dZ1, "dW1": dW1, "db1": db1}
    return gradients
```

```
# Updating the weights based on the negative gradients
def updateParameters(parameters, gradients, learningRate):
    parameters["W1"] = parameters["W1"] - learningRate * gradients["dW1"]
    parameters["W2"] = parameters["W2"] - learningRate * gradients["dW2"]
    parameters["b1"] = parameters["b1"] - learningRate * gradients["db1"]
    parameters["b2"] = parameters["b2"] - learningRate * gradients["db2"]
    return parameters

# Model to learn the AND truth table
X = np.array([[0, 0, 1, 1], [0, 1, 0, 1]]) # AND input
Y = np.array([[0, 0, 0, 1]]) # AND output

# Define model parameters
neuronsInHiddenLayers = 2 # number of hidden layer neurons (2)
inputFeatures = X.shape[0] # number of input features (2)
outputFeatures = Y.shape[0] # number of output features (1)
parameters = initializeParameters(inputFeatures, neuronsInHiddenLayers, outputFeatures)
epoch = 100000
learningRate = 0.01
losses = np.zeros((epoch, 1))

for i in range(epoch):
    losses[i, 0], cache, A2 = forwardPropagation(X, Y, parameters)
    gradients = backwardPropagation(X, Y, cache)
    parameters = updateParameters(parameters, gradients, learningRate)

# Evaluating the performance
plt.figure()
plt.plot(losses)
plt.xlabel("EPOCHS")
plt.ylabel("Loss value")
plt.show()

# Testing
X = np.array([[1, 1, 0, 0], [0, 1, 0, 1]]) # AND input
cost, _, A2 = forwardPropagation(X, Y, parameters)
prediction = (A2 > 0.5) * 1.0
# print(A2)
print(prediction)

# OUTPUT :
# [[ 0.  1.  0.  0.]]



#OR operation

# import Python Libraries
import numpy as np
from matplotlib import pyplot as plt

# Sigmoid Function
def sigmoid(z):
    return 1 / (1 + np.exp(-z))

# Initialization of the neural network parameters
# Initialized all the weights in the range of between 0 and 1
# Bias values are initialized to 0
def initializeParameters(inputFeatures, neuronsInHiddenLayers, outputFeatures):
    W1 = np.random.randn(neuronsInHiddenLayers, inputFeatures)
```

```
    W2 = np.random.randn(outputFeatures, neuronsInHiddenLayers)
    b1 = np.zeros((neuronsInHiddenLayers, 1))
    b2 = np.zeros((outputFeatures, 1))

    parameters = {"W1" : W1, "b1": b1,
                  "W2" : W2, "b2": b2}
    return parameters

# Forward Propagation
def forwardPropagation(X, Y, parameters):
    m = X.shape[1]
    W1 = parameters["W1"]
    W2 = parameters["W2"]
    b1 = parameters["b1"]
    b2 = parameters["b2"]

    Z1 = np.dot(W1, X) + b1
    A1 = sigmoid(Z1)
    Z2 = np.dot(W2, A1) + b2
    A2 = sigmoid(Z2)

    cache = (Z1, A1, W1, b1, Z2, A2, W2, b2)
    logprobs = np.multiply(np.log(A2), Y) + np.multiply(np.log(1 - A2), (1 - Y))
    cost = -np.sum(logprobs) / m
    return cost, cache, A2

# Backward Propagation
def backwardPropagation(X, Y, cache):
    m = X.shape[1]
    (Z1, A1, W1, b1, Z2, A2, W2, b2) = cache

    dZ2 = A2 - Y
    dW2 = np.dot(dZ2, A1.T) / m
    db2 = np.sum(dZ2, axis = 1, keepdims = True)

    dA1 = np.dot(W2.T, dZ2)
    dZ1 = np.multiply(dA1, A1 * (1- A1))
    dW1 = np.dot(dZ1, X.T) / m
    db1 = np.sum(dZ1, axis = 1, keepdims = True) / m

    gradients = {"dZ2": dZ2, "dW2": dW2, "db2": db2,
                 "dZ1": dZ1, "dW1": dW1, "db1": db1}
    return gradients

# Updating the weights based on the negative gradients
def updateParameters(parameters, gradients, learningRate):
    parameters["W1"] = parameters["W1"] - learningRate * gradients["dW1"]
    parameters["W2"] = parameters["W2"] - learningRate * gradients["dW2"]
    parameters["b1"] = parameters["b1"] - learningRate * gradients["db1"]
    parameters["b2"] = parameters["b2"] - learningRate * gradients["db2"]
    return parameters

# Model to learn the OR truth table
X = np.array([[0, 0, 1, 1], [0, 1, 0, 1]]) # OR input
Y = np.array([[0, 1, 1, 1]]) # OR output

# Define model parameters
neuronsInHiddenLayers = 2 # number of hidden layer neurons (2)
inputFeatures = X.shape[0] # number of input features (2)
outputFeatures = Y.shape[0] # number of output features (1)
```

```
parameters = initializeParameters(inputFeatures, neuronsInHiddenLayers, outputFeatures)
epoch = 100000
learningRate = 0.01
losses = np.zeros((epoch, 1))

for i in range(epoch):
    losses[i, 0], cache, A2 = forwardPropagation(X, Y, parameters)
    gradients = backwardPropagation(X, Y, cache)
    parameters = updateParameters(parameters, gradients, learningRate)

# Evaluating the performance
plt.figure()
plt.plot(losses)
plt.xlabel("EPOCHS")
plt.ylabel("Loss value")
plt.show()

# Testing
X = np.array([[1, 1, 0, 0], [0, 1, 0, 1]]) # OR input
cost, _, A2 = forwardPropagation(X, Y, parameters)
prediction = (A2 > 0.5) * 1.0
# print(A2)
print(prediction)

#OUTPUT :
# [[ 1.  1.  0.  1.]]
```



Experiment No. 4

1. **Aim:** Implementation of Single layer Perceptron Learning Algorithm.

2. **Objectives:**
   - To become familiar with neural networks learning algorithms from available examples.
   - Provide knowledge of learning algorithm in neural networks.

3. **Outcomes:** The student will be able to,
   - Have an understanding of the concepts and techniques of neural networks through the study of the most important neural network models**.**
   - Discuss the main factors involved in achieving good learning and generalization performance in neural network systems.
   - Use the current techniques and tools required for computing practice.

4. **Software Required:** JAVA / MATLAB/OCTAVE

5. **Theory:**

Neural networks are a branch of ―Artificial Intelligence". Artificial Neural Network is a system loosely modelled based on the human brain. Neural networks are a powerful technique to solve many real world problems. They have the ability to learn from experience in order to improve their performance and to adapt themselves to changes in the environment. In addition to that they are able to deal with incomplete information or noisy data and can be very effective especially in situations where it is not possible to define the rules or steps that lead to the solution of a problem. In a nutshell a Neural network can be considered as a black box that is able to predict an output pattern when it recognizes a given input pattern. Once trained, the neural network is able to recognize similarities when presented with a new input pattern, resulting in a predicted output pattern.

In late 1950s, Frank Rosenblatt introduced a network composed of the units that were enhanced version of McCulloch-Pitts Threshold Logic Unit (TLU) model. Rosenblatt's model of neuron, a perceptron, was the result of merger between two concepts from the 1940s, McCulloch-Pitts model of an artificial neuron and Hebbian learning rule of adjusting weights. In addition to the variable weight values, the perceptron model added an extra input that represents bias. Thus, the modified equation is now as follows:

Where b represents the bias value.

6. Algorithm:

Perceptron Learning Algorithm:

The perceptron learning rule was originally developed by Frank Rosenblatt in the late 1950s. Training patterns are presented to the network's inputs; the output is computed. Then the connection weights$w_j$are modified by an amount that is proportional to the product of

- the difference between the actual output, y, and the desired output, d, and
- the input pattern, x.

The algorithm is as follows:

1. Initialize the weights and threshold to small random numbers.
2. Present a vector x to the neuron inputs and calculate the output.
3. Update the weights according to:

$$w_j(t+1) = w_j(t)+\eta(d-y)x$$

where

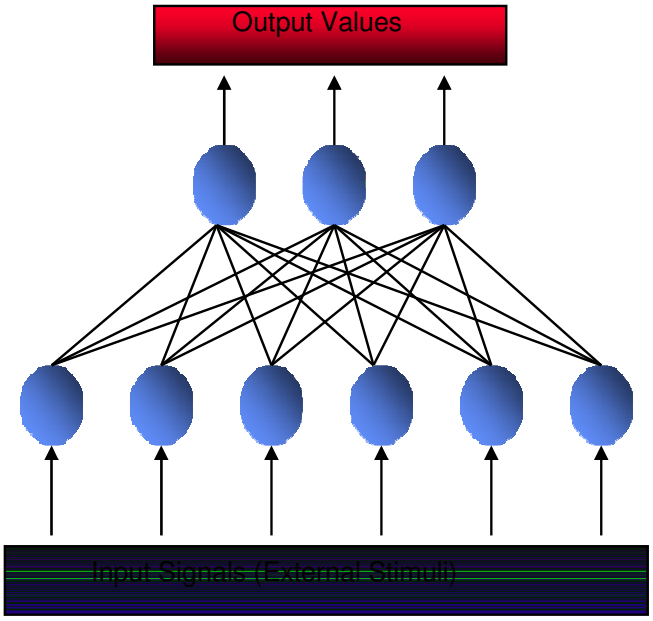
- d is the desired output,
- t is the iteration number, and
- eta is the gain or step size, where 0.0 < n < 1.0

4. Repeat steps 2 and 3 until:

    1. the iteration error is less than a user-specified error threshold or
    2. a predetermined number of iterations have been completed.

Learning only occurs when an error is made; otherwise the weights are left unchanged.

Multilayer Perceptron



Output Layer

Adjustable Weights

Input Layer

Problem Statement: Implement AND function using perceptron model

Truth table for AND function is:

X1	X2	Y
0	0	0
0	1	0
1	0	0
1	1	1

7. Conclusion:

Single layer perceptron learning algorithm is implemented for AND function. It is used for train the iterations of neural network. Neural network mimics the human brain and perceptron learning algorithm trains the neural network according to the input given.

8. Viva Questions:

- What is feed forward network?
- Write the logistic sigmoid function?
- Why use Artificial Neural Networks? What are its advantages?
- List some commercial practical applications of Artificial Neural Networks.
- What are the disadvantages of Artificial Neural Networks?

```
%Perceptron for AND funtion
clear;
clc;
x=[1 1 -1 -1;1 -1 1 -1];
t=[1 -1 -1 -1];
w=[0 0];
b=0;
alpha=input('Enter Learning rate=');
theta=input('Enter Threshold value=');
con=1;
epoch=0;
while con
con=0;
for i=1:4
yin=b+x(1,i)*w(1)+x(2,i)*w(2);
if yin>theta
y=1;
end
if yin <=theta & yin>=-theta
y=0;
end
if yin<-theta
y=-1;
end
if y-t(i)
con=1;
for j=1:2
w(j)=w(j)+alpha*t(i)*x(j,i);
end
```

```
b=b+alpha*t(i);
end
end
epoch=epoch+1;
end
disp('Perceptron for AND function');
disp(' Final Weight matrix');
disp(w);
disp('Final Bias');
disp(b);
```

Output

```
Enter Learning rate=1
Enter Threshold value=0.5
Perceptron for AND function
Final Weight matrix
1 1
Final Bias
-1
```

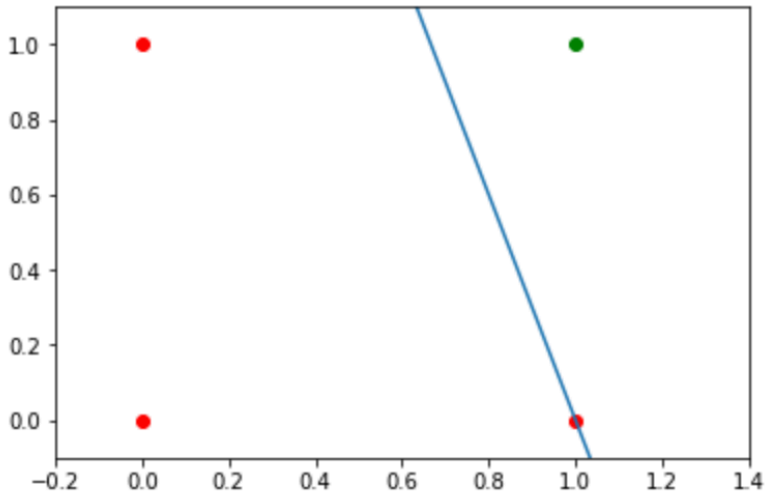
CODE :

```
import numpy as np
from collections import Counter

class Perceptron:

    def __init__(self,
                 weights,
                 bias=1,
                 learning_rate=0.3):
        """
        'weights' can be a numpy array, list or a tuple with the
        actual values of the weights. The number of input values
        is indirectly defined by the length of 'weights'
        """
        self.weights = np.array(weights)
        self.bias = bias
        self.learning_rate = learning_rate

    @staticmethod
    def unit_step_function(x):
        if  x <= 0:
            return 0
        else:
            return 1

    def __call__(self, in_data):
        in_data = np.concatenate( (in_data, [self.bias]) )
        result = self.weights @ in_data
        return Perceptron.unit_step_function(result)

    def adjust(self,
               target_result,
               in_data):
        if type(in_data) != np.ndarray:
            in_data = np.array(in_data)  #
        calculated_result = self(in_data)
        error = target_result - calculated_result
        if error != 0:
            in_data = np.concatenate( (in_data, [self.bias]) )
            correction = error * in_data * self.learning_rate
            self.weights += correction

    def evaluate(self, data, labels):
        evaluation = Counter()
        for sample, label in zip(data, labels):
            result = self(sample) # predict
            if result == label:
                evaluation["correct"] += 1
            else:
                evaluation["wrong"] += 1
        return evaluation

import numpy as np
from perceptrons import Perceptron

def labelled_samples(n):
    for _ in range(n):
        s = np.random.randint(0, 2, (2,))
```

```
        yield (s, 1) if s[0] == 1 and s[1] == 1 else (s, 0)

p = Perceptron(weights=[0.3, 0.3, 0.3],
               learning_rate=0.2)

for in_data, label in labelled_samples(30):
    p.adjust(label,
             in_data)

test_data, test_labels = list(zip(*labelled_samples(30)))

evaluation = p.evaluate(test_data, test_labels)
print(evaluation)

# Counter({'correct': 30})

import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots()
xmin, xmax = -0.2, 1.4
X = np.arange(xmin, xmax, 0.1)
ax.scatter(0, 0, color="r")
ax.scatter(0, 1, color="r")
ax.scatter(1, 0, color="r")
ax.scatter(1, 1, color="g")
ax.set_xlim([xmin, xmax])
ax.set_ylim([-0.1, 1.1])
m = -p.weights[0] / p.weights[1]
c = -p.weights[2] / p.weights[1]
print(m, c)
ax.plot(X, m * X + c )
plt.plot()
```

OUTPUT:

-3.0000000000000004 ,3.0000000000000013



Experiment No. 5

1. **Aim:** Study and Implement optimization problem using Particle Swarm Optimization
2. **Objectives:**
    - To become familiar with optimization methods.
    - To get acquainted with implementation of Partial Swarm Optimization.
3. **Outcomes:** The student will be able to,
    - Understand the optimization method, PSO.
4. **Software Required: C/ C++/**JAVA/ MATLAB
5. **Theory:**

1. Introduction

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling (Schools are groups of *fish* that act as a single unit, and are characterized by a streamlined structure and uniform behavior for the purposes of avoiding predators and finding food.).

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called **particles**, fly through the problem space by following the current optimum particles.

Compared to GA, the **advantages of PSO** are that PSO is easy to implement and there are few parameters to adjust. PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where GA can be applied.

2. Background: Artificial life

The term "Artificial Life" (ALife) is used to describe research into human-made systems that possess some of the essential properties of life. ALife includes two-folded research topic:

1. ALife studies how computational techniques can help when studying biological phenomena
2. ALife studies how biological techniques can help out with computational problems

Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial. The concept is employed in work on artificial intelligence. The expression was introduced by Gerardo Beni and Jing Wang in 1989, in the context of cellular robotic systems.

There are two popular swarm inspired methods in computational intelligence areas: **Ant colony optimization (ACO) and particle swarm optimization (PSO).** ACO was inspired by the behaviors of ants and has many successful applications in discrete optimization problems.

The particle swarm concept originated as a simulation of simplified social system. The original intent was to graphically simulate the choreography of bird of a bird block or fish school. However, it was found that particle swarm model can be used as an optimizer.

3. The algorithm

As stated before, PSO simulates the behaviors of bird flocking. Suppose the following scenario: a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is. But they know how far the food is in each iteration. So what's the best strategy to find the food? The effective one is to follow the bird which is nearest to the food.

PSO learned from the scenario and used it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. We call it "particle". All of particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

Global Optimization

A global minimizer is defined as x^* such that

$$f(x^*) \leq f(x), \ \forall x \in S \qquad (2.4)$$

where S is the search space and $S = R^n$ for unconstrained problems.

Here, the term global minimum refers to the value $f(x^*)$, and x^* is called the global minimizer. Some global optimization methods require a starting point $z_0 \in S$ and it will be able to find the global minimizer x^* if $z_0 \in S$.

Local Optimization

A local minimizer x_L^* of the region L, is defined as

$$f(x_L^*) \leq f(x), \ \forall x \in L \qquad (2.5)$$

where $L \subseteq R^n$.

Here, a local optimization method should guarantee that a local minimizer of the set L is found.

For gbest PSO method, the velocity of particle i is calculated by

$$v_{ij}^{t+1} = v_{ij}^{t} + c_1 r_{1j}^{t}\left[P_{best,i}^{t} - x_{ij}^{t}\right] + c_2 r_{2j}^{t}\left[G_{best} - x_{ij}^{t}\right] \quad (3.7)$$

where

v_{ij}^{t} is the velocity vector of particle i in dimension j at time t;

x_{ij}^{t} is the position vector of particle i in dimension j at time t;

$P_{best,i}^{t}$ is the personal best position of particle i in dimension j found from initialization through time t;

G_{best} is the global best position of particle i in dimension j found from initialization through time t;

c_1 and c_2 are positive acceleration constants which are used to level the contribution of the cognitive and social components respectively;

r_{1j}^{t} and r_{2j}^{t} are random numbers from uniform distribution $U(0,1)$ at time t.

$$v_{ij}^{t+1} = v_{ij}^{t} + c_1 r_{1j}^{t}\left[P_{best,i}^{t} - x_{ij}^{t}\right] + c_2 r_{2j}^{t}\left[L_{best,i} - x_{ij}^{t}\right] \quad (3.8)$$

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called **pbest**. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called **gbest**. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called **lbest**.
After finding the two best values, the particle updates its velocity and positions with following equation (a) and (b).

v[] = v[] + c1 * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[]) (a)
present[] = persent[] + v[] (b)

v[] is the particle velocity, persent[] is the current particle (solution). pbest[] and gbest[] are defined as stated before. rand () is a random number between (0,1). c1, c2 are learning factors. usually c1 = c2 = 2.

The pseudo code of the procedure is as follows

```
For each particle
   Initialize particle
END
Do
   For each particle
      Calculate fitness value
      If the fitness value is better than the best fitness value (pBest) in history
         set current value as the new pBest
```

End
Choose the particle with the best fitness value of all the particles as the gBest
For each particle
Calculate particle velocity according equation (a)
Update particle position according equation (b)
End
While maximum iterations or minimum error criteria is not attained

4. Comparisons between Genetic Algorithm and PSO

Most of evolutionary techniques have the following procedure:

1. Random generation of an initial population
2. Reckoning of a fitness value for each subject. It will directly depend on the distance to the optimum.
3. Reproduction of the population based on fitness values.
4. If requirements are met, then stop. Otherwise go back to 2.

From the procedure, we can learn that PSO shares many common points with GA. Both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population. Both update the population and search for the optimium with random techniques. **Both systems do not guarantee success.**

However, PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which is important to the algorithm. Compared with genetic algorithms (GAs), the information sharing mechanism in PSO is significantly different. In GAs, chromosomes share information with each other. So the whole population moves like a one group towards an optimal area. In PSO, only gBest (or lBest) gives out the information to others. It is a one -way information sharing mechanism. The evolution only looks for the best solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases.

5. Artificial neural network and PSO

An artificial neural network (ANN) is an analysis paradigm that is a simple model of the brain and the back-propagation algorithm is the one of the most popular method to train the artificial neural network. Recently there have been significant research efforts to apply evolutionary computation (EC) techniques for the purposes of evolving one or more aspects of artificial neural networks.

Evolutionary computation methodologies have been applied to three main attributes of neural networks: network connection weights, network architecture (network topology, transfer function), and network learning algorithms. Most of the work involving the evolution of ANN has focused on the network weights and topological structure. Usually the weights and/or topological structure are encoded as a chromosome in GA. The selection of fitness function depends on the research goals. For a classification problem, the rate of mis-classified patterns can be viewed as the fitness value.

One can compare a simple example of evolving ANN with PSO. **The problem is a benchmark function of classification problem: iris data set.** Measurements of four attributes of iris flowers are provided in each data set record: sepal length, sepal width, petal length, and petal width. Fifty sets of measurements are present for each of three varieties of iris flowers, for a total of 150 records, or patterns.

A 3-layer ANN is used to do the classification. There are 4 inputs and 3 outputs. So the input layer has 4 neurons and the output layer has 3 neurons. One can evolve the number of hidden neurons. Suppose the hidden layer has 6 neurons. We can evolve other parameters in the feed-forward network. Here we only evolve the network weights. So the particle will be a group of weights, there are 4*6+6*3 = 42 weights, so **the particle consists of 42 real numbers**. The range of weights can be set to [-100, 100] (this is just a example, in real cases, one might try different ranges). After encoding the particles, we need to determine the fitness function. For the classification problem, we feed all the patterns to the network whose weights is determined by the particle, get the outputs and compare it the standard outputs. Then we record the number of misclassified patterns as the fitness value of that particle. Now we can apply PSO to train the ANN to get lower number of misclassified patterns as possible. There are not many parameters in PSO need to be adjusted. We only need to adjust the number of hidden layers and the range of the weights to get better results in different trials.

6. PSO parameter control

From the above case, we can learn that there are two key steps when applying PSO to optimization problems: the representation of the solution and the fitness function. One of the advantages of PSO is that PSO take real numbers as particles. It is not like GA, which needs to change to binary encoding, or special genetic operators have to be used. For example, we try to find the solution for f(x) = x1^2 + x2^2+x3^2, the particle can be set as (x1, x2, x3), and fitness function is f(x). Then we can use the standard procedure to find the optimum. The searching is a repeat process, and the stop criteria are that the maximum iteration number is reached or the minimum error condition is satisfied.

Dimension of particles: It is determined by the problem to be optimized,
Range of particles: It is also determined by the problem to be optimized, you can specify different ranges for different dimension of particles. The number of particles: the typical range is 20 - 40. Actually for most of the problems 10 particles is large enough to get good results.

Vmax: it determines the maximum change one particle can take during one iteration. Usually we set the range of the particle as the Vmax for example, the particle (x1, x2, x3)
X1 belongs [-10, 10], then Vmax = 20

•When $c_1 = c_2 = 0$, then all particles continue flying at their current speed until they hit the search space's boundary. Therefore, from the equations (3.7) and (3.8), the velocity update equation is calculated as

$$v_{ij}^{t+1} = v_{ij}^{t}$$

- When $c_1 > 0$ and $c_2 = 0$, all particles are independent. The velocity update equation will be

$$v_{ij}^{t+1} = v_{ij}^{t} + c_1 r_{1j}^{t}\left[P_{best,i}^{t} - x_{ij}^{t}\right]$$

On the contrary, when $c_2 > 0$ and $c_1 = 0$, all particles are attracted to a single point ($i.e.\ G_{best}$) in the entire swarm and the update velocity will become

$$v_{ij}^{t+1} = v_{ij}^{t} + c_2 r_{2j}^{t}\left[G_{best} - x_{ij}^{t}\right] \text{ for gbest PSO.}$$

$$\text{or, } v_{ij}^{t+1} = v_{ij}^{t} + c_2 r_{2j}^{t}\left[L_{best,i} - x_{ij}^{t}\right] \text{ for lbest PSO.}$$

- When $c_1 = c_2$, all particles are attracted towards the average of $P_{best,i}^{t}$ and G_{best}.

The stop condition: the maximum number of iterations the PSO execute and the minimum error requirement. This stop condition depends on the problem to be optimized.

7. Example - Problem Formulation of PSO Algorithm:

Problem formulation of PSO Algo.

Problem :- Find maximum of function
$f(x) = -x^2 + 5x + 20$ with $-10 \le x \le 10$
Use PSO Algo. with 9 particles with initial positions $x_1 = -9.6$, $x_2 = -6$, $x_3 = -2.6$
$x_4 = -1.1$, $x_5 = 0.6$, $x_6 = 2.3$, $x_7 = 2.8$, $x_8 = 8.3$, $x_9 = 10$.

Solution :-

Step-1 :- The initial population (ie. $t=0$) represent as x_i^t.
x_i^0 ; $i = 1, 2, \cdots, 9$
Evaluate objective fun. as f_i^t
$f_1^0 = -120$, $f_2^0 = -46$, $f_3^0 = 0.24$, $f_4^0 = 13.29$, $f_5^0 = 22.64$
$f_6^0 = 26.21$, $f_7^0 = 26.16$, $f_8^0 = -7.39$, $f_9^0 = -30$
Let $c_1 = c_2 = 1$.
Let initial velocity of particle is zero, v_i^t
$v_i^0 = 0$ for $\forall i$

Step2 $t = t + 1$,

CODE :

```
from __future__ import division
import random
import math

# function we are attempting to optimize (minimize)
def func1(x):
    total=0
    for i in range(len(x)):
        total+=x[i]**2
    return total

class Particle:
    def __init__(self,x0):
        self.position_i=[]          # particle position
        self.velocity_i=[]          # particle velocity
        self.pos_best_i=[]          # best position individual
        self.err_best_i=-1          # best error individual
        self.err_i=-1               # error individual

        for i in range(0,num_dimensions):
            self.velocity_i.append(random.uniform(-1,1))
            self.position_i.append(x0[i])

    # evaluate current fitness
    def evaluate(self,costFunc):
        self.err_i=costFunc(self.position_i)

        # check to see if the current position is an individual best
        if self.err_i < self.err_best_i or self.err_best_i==-1:
            self.pos_best_i=self.position_i
            self.err_best_i=self.err_i

    # update new particle velocity
    def update_velocity(self,pos_best_g):
        w=0.5       # constant inertia weight (how much to weigh the previous velocity)
        c1=1        # cognative constant
        c2=2        # social constant

        for i in range(0,num_dimensions):
            r1=random.random()
            r2=random.random()

            vel_cognitive=c1*r1*(self.pos_best_i[i]-self.position_i[i])
            vel_social=c2*r2*(pos_best_g[i]-self.position_i[i])
            self.velocity_i[i]=w*self.velocity_i[i]+vel_cognitive+vel_social

    # update the particle position based off new velocity updates
    def update_position(self,bounds):
        for i in range(0,num_dimensions):
            self.position_i[i]=self.position_i[i]+self.velocity_i[i]

            # adjust maximum position if necessary
            if self.position_i[i]>bounds[i][1]:
                self.position_i[i]=bounds[i][1]

            # adjust minimum position if neseccary
            if self.position_i[i] < bounds[i][0]:
                self.position_i[i]=bounds[i][0]
```

```
class PSO():
    def __init__(self,costFunc,x0,bounds,num_particles,maxiter):
        global num_dimensions

        num_dimensions=len(x0)
        err_best_g=-1                   # best error for group
        pos_best_g=[]                   # best position for group

        # establish the swarm
        swarm=[]
        for i in range(0,num_particles):
            swarm.append(Particle(x0))

        # begin optimization loop
        i=0
        while i < maxiter:
            #print i,err_best_g
            # cycle through particles in swarm and evaluate fitness
            for j in range(0,num_particles):
                swarm[j].evaluate(costFunc)

                # determine if current particle is the best (globally)
                if swarm[j].err_i < err_best_g or err_best_g == -1:
                    pos_best_g=list(swarm[j].position_i)
                    err_best_g=float(swarm[j].err_i)

            # cycle through swarm and update velocities and position
            for j in range(0,num_particles):
                swarm[j].update_velocity(pos_best_g)
                swarm[j].update_position(bounds)
            i+=1

        # print final results
        print ('FINAL:')
        print (pos_best_g)
        print (err_best_g)

if __name__ == "__PSO__":
    main()

#--- RUN ----------------------------------------------------------------------+

initial=[5,5]               # initial starting location [x1,x2...]
bounds=[(-10,10),(-10,10)]  # input bounds [(x1_min,x1_max),(x2_min,x2_max)...]
PSO(func1,initial,bounds,num_particles=15,maxiter=30)
```

#**OUTPUT** :

```
# FINAL:
# [0.0016146750800351108, 0.0005647507298002629]
# 2.926119000896321e-06
```

Sinhgad College of Engineering Department of Computer Engineering

Name of Student: **Prashant Kumar**

Roll No: **405B049**

PRN Number: **71813716H**

Seat No: **B150234347**

Class: **BE** Div.: **2** Batch: B

Subject: **Cloud Computing**

Name of Laboratory: **Laboratory Practice IV**

List of Assignments

S.NO.	Title of Assignment	Remark	Signature
1	Installation and configuration of own Cloud		
2	Implementation of Virtualization in Cloud Computing to Learn Virtualization Basics, Benefits of Virtualization in Cloud using Open Source Operating System.		
3	Case study on Amazon EC2 to learn about Amazon EC2, Amazon Elastic Compute Cloud is a central part of amazon.com's cloud computing platform, Amazon Web Services. How EC2 allows users torrent virtual computers on which to run their own computer applications.		
4	Case study on Microsoft azure to learn about Microsoft Azure is a cloud computing platform and infrastructure, created by Microsoft, for building, deploying and managing applications and services through a global network of Microsoft-managed data centre's. How it works, different services provided by it.		
5	Write a Program to Create, Manage and groups User accounts in own Cloud by Installing Administrative Features.		
6	Assignment to install and configure Google App Engine.		

Page No.
Date

Assignment No. 1

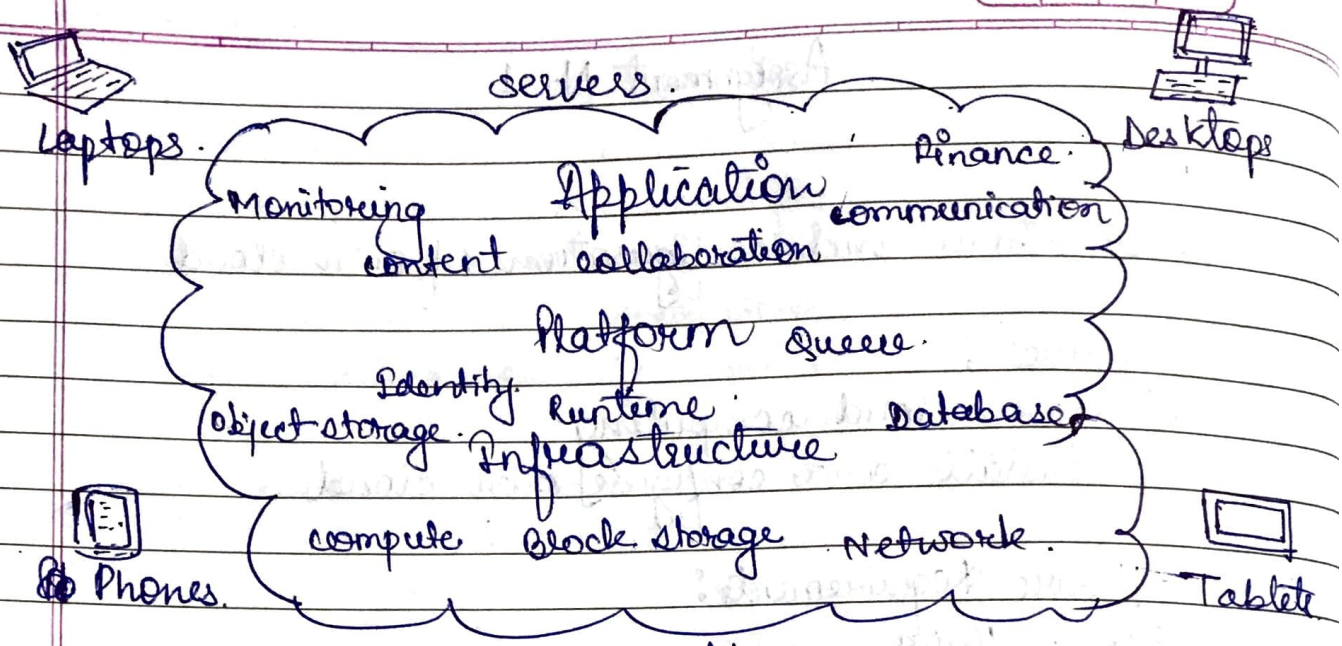
Aim:
Installation and configuration of own cloud.

Objectives:
1. To learn cloud computing.
2. To install and configure own cloud.

Software Requirements:
Ubuntu 16.04
PHP
MySQL.

Hardware Requirements:
Pentium IV system with latest configuration.

Theory:
Cloud computing is a method of delivering information Technology (IT) services in which resources are retrieved from the Internet through web-based tools and applications, as opposed to a direct connection to a server. Rather than keeping files on a proprietary hard drive or local storage service, cloud-based storage makes it possible to save them to a remote database. As long as an electronic device has access to the web, it has access to the data and the software programs to run it.



Cloud computing

Cloud Computing — Types of Cloud.

Cloud computing is usually described in one of two ways. Either based on the deployment model, or on the service that the cloud is offering. Based on deployment model, we can classify cloud as

- Public
- Private
- Hybrid
- Community cloud.

Based on a service the cloud model is offering, we are speaking of either:

- IaaS (Infrastructure-as-a-service)
- PaaS (Platform-as-a-service)
- SaaS (software-as-a-service)

Ore, storage, Database, Information, Process, Applicatio Integration, Security, Management, Testing-as-a-servic

Page No.
Date

Basically, programs that are needed to run a certain applications are now more popularly located on a remote machine, owned by another company. This is done in order not to loose on the quality performance due to processing power of your own company computer, to save money on IT support, and yet remain advantageous on the market. These computers that run the applications, store the data, and use a server system, are basically what we call "the cloud".

Public Cloud.

When we talk about public cloud, we mean that the whole computing infrastructure is located on the premises of a cloud computing company that offers the cloud service. The location remains there, separate from the customer and he has no physical control over the infrastructure.

As public clouds use shared resources, they do excel mostly in performance, but are also most vulnerable to various attacks.

GlobalDots offers worldwide Public cloud service is leading data centers. Our experts will assist you in choosing the right solution for you.

Private Cloud.

Private cloud provides the same benifits of Public cloud, but uses dedicated, private hardware. Private cloud means using a cloud Infrastructure (network) solely by one customer/organization. It is not shared with others, yet it is remotely located. The companies have an option of choosing an

On-premise private cloud as well, which is more expensive, but they do have a physical control over the infrastructure.
The security and control level is highest while using a private network. Yet, the cost reduction can be minimal, if the company needs to invest in an on-premise cloud infrastructure.

GlobalDots offers worldwide private cloud service in leading data centers.

With our private cloud you'll get:

- Increased redundancy.
- Decreased provising time for new servers.
- Saved capital by eliminating hardware support contracts.
- Quicker expendability compared to hosting your own physical servers.
- Use of dedicated, private hardware.

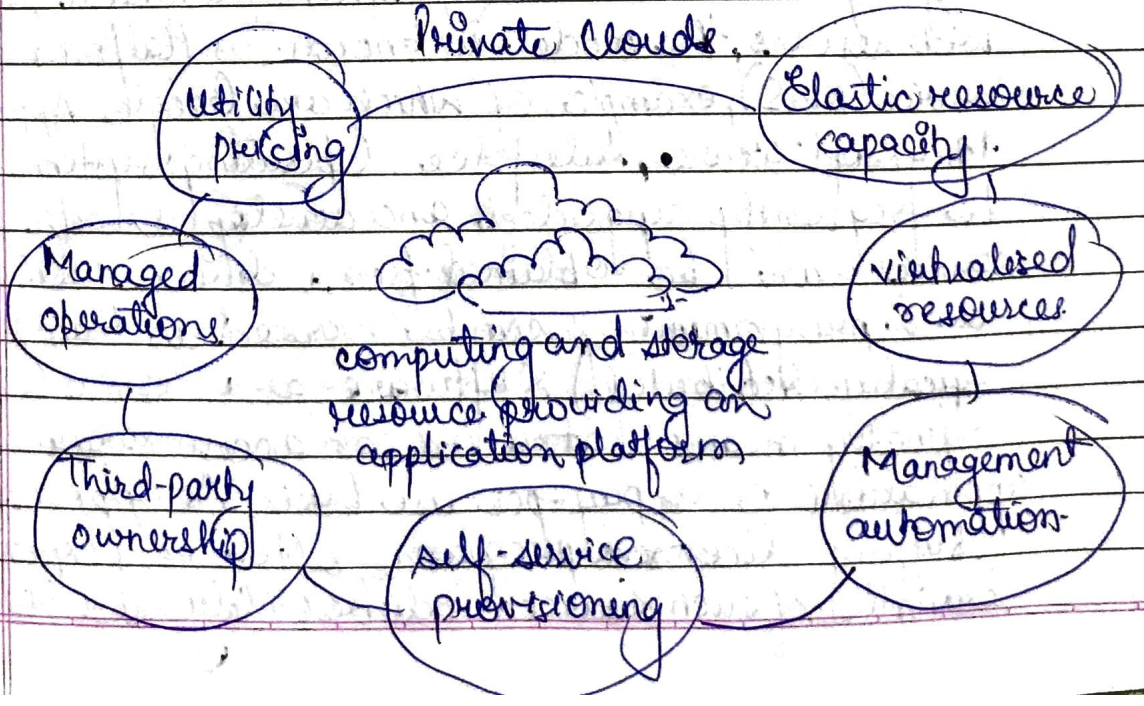
Hybrid cloud.

Hybrid cloud, of course, means using both private and public clouds, depending on their purpose. For example, public cloud can be used to interact with customers, while keeping their data secured through a private cloud. Most people associate traditional public cloud services with elastic, scalibility and the ability to handle constant shifts in demand. However, performance issue can arise for certain data-intensive or high-availability workloads.

GlobalDots offer combines hybrid cloud with bare-metal and virtualized clouds into a unified environment allowing your business to optimize for scale, performance and cost simultaneously.

Public Clouds.

Private Clouds.

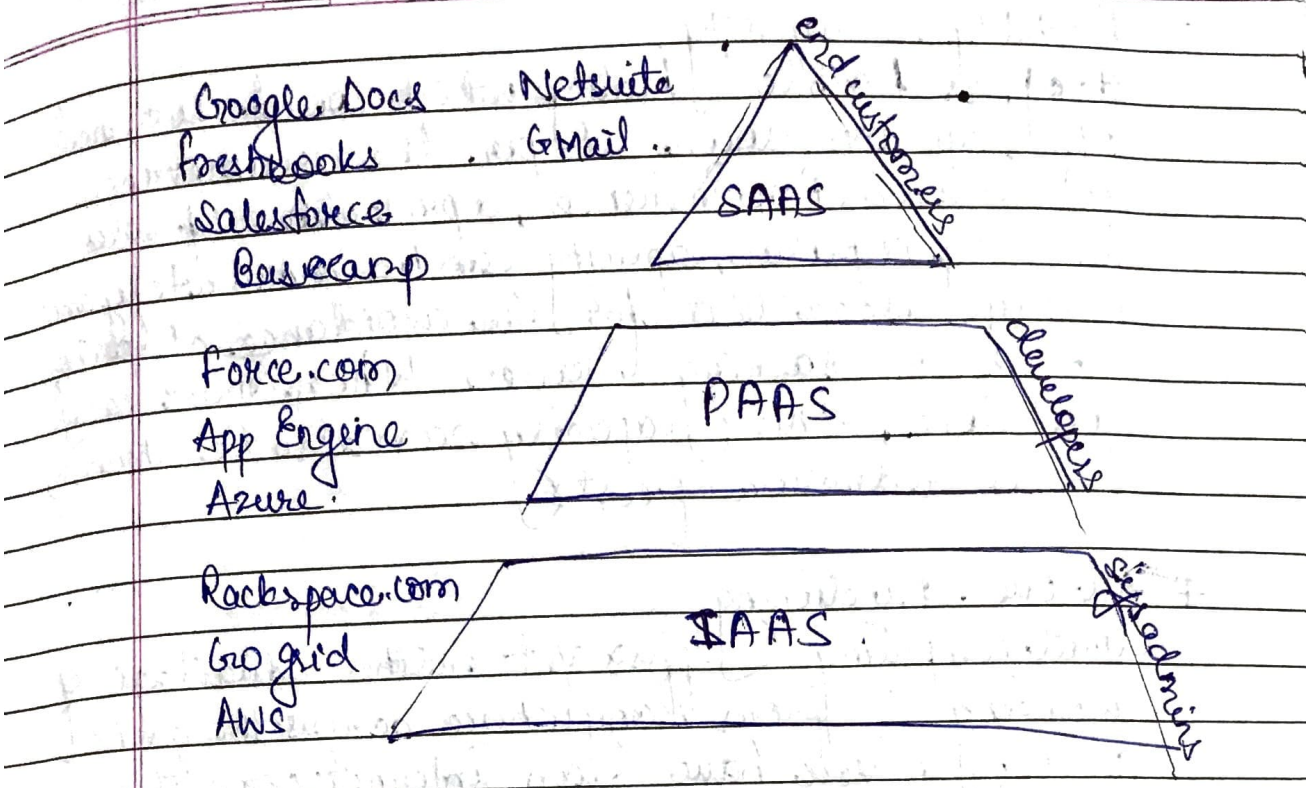


Page No.
Data

Community cloud.
It implies an infrastructure that is shared between organisations usually with the shared data and data management concerns. For example, a community cloud can belong to a government of a single country. Community clouds can be located both on and off the premises.

The most popular services of the cloud are that of either infrastructure, platform, software or storage.
As explained before, the most common cloud services is that one offering data storage disks and virtual servers, i.e, infrastructure. Examples of Infrastructure-as-a-service (IaaS) companies are Amazon, Rackspace, Flexiscale.

If the cloud offers a development platform, and this includes Operating system, programming language execution environment, database, and web servers, the model is known as Platform-as-a-Service (PaaS), example of which are Google App Engine, Microsoft azure, Salesforce. Operating system can be frequently upgraded and developed with PaaS, services can be obtained from diverse sources and programming can be worked in teams (geographically distributed). Software-as-a-service (SaaS), finally means that users can access various software applications on a pay-per-use basis. As opposed to buying licensed programs, often very expensive. Examples of such services include widely used GMail



Advantages of Cloud computing

1. Less costs.
The service are free from capital expenditure. There are no huge costs of hardware in cloud computing. You just have to pay as you operate it and enjoy the model based on your subscription plan.

2. 24×7 Availability.
Most of the cloud providers are truly reliable in offering their services, with most of them maintaining Uptime of 99.9%. The workers can get onto the applications needed basically from anywhere. Some of the functions even function off-line.

3. Flexibility in Capacity
It offers flexible facility which could be turned off up or down as per the circumstance of the user. For instance, a promotions of sales is very popular, capacity can be immediately and quickly added to it for the avoidance of losing sales and crashing servers. When those sales are done, the capacity can also be shrunk for the reduction of costs.

4. All over Functioning.
Cloud computing offers yet another advantage of working from anywhere across the globe, as long as you have an internet connections. Even while using the critical cloud services that offers mobile apps, there is no limitations of the device used.

5. Automated update on Software.
In cloud computing, the server supplier regularly update your software including the updates on security, so that you do not need to agonize on wasting your crucial time on maintaing the system. You find extra time to focus on the important thing like, How to grow your business.

6. Security
Cloud computing offers great security when any sensitive data has been lost. As the data is stored in the system, it can be easily accessed even if something happens to your computer. You

can even remotely wipe out data from the lost machines for avoiding it getting in the wrong hands.

7. Carbon Footprint.
Cloud computing is helping out organization to reduce their carbon footprint. Organisations utilize only the amount of resources they need, which helps them to avoid any over-prowsing. Hence, no waste of resources (and thus energy).

8. Enhanced collaboration.
Cloud applications enhance collaboration by authorizing diverse groups of people virtually meet and exchange information with the help of shared storage. Such capability helps is improving the customer service and product development and also reducing the marketing time.

9. Control on the Documents.
Before cloud came into being, workers needed to send files in and out as the email attachments for being worked on by a single user at one time ultimately ending up with a mess of contrary titles, formats and file content. Moving to cloud computing has facilitated central file storage.

10. Easily Manageable.
Cloud computing offers simplified and enhanced IT maintenance and management capabilities by

agreement backed by SLA, central resource administration and managed infrastructure. You get to enjoy a basic user interface without any requirement for installation. Plus you are assured guaranteed and timely management, maintenance, and delivery of the IT services.

Applications of cloud computing :-

1. Online file storage.
2. Photo editing software.
3. Digital video software.
4. Twitter-related applications.
5. Creating image album.
6. Web application for antivirus.
7. Word processing application.
8. Spreadsheet.
9. Presentation application.
10. Finding a way on the map.
11. Ecomerce software.
12. Miscellaneous application.

Open Terminal and type :-

1. sudo apt update && sudo apt upgrade.
2. sudo wget -nv -O Release.key
3. Sudo apt-key add - < Release.key
4. Sud. sh-c "echo 'deb http://download.owncloud.org/download/repositories/9.1/Ubuntu_16.04/ /' > /etc/apt/source.list.d/owncloud.list".
5. Sudo apt update
6. Sudo apt install owncloud.

Open another terminal
mysql -u root -p
password : root

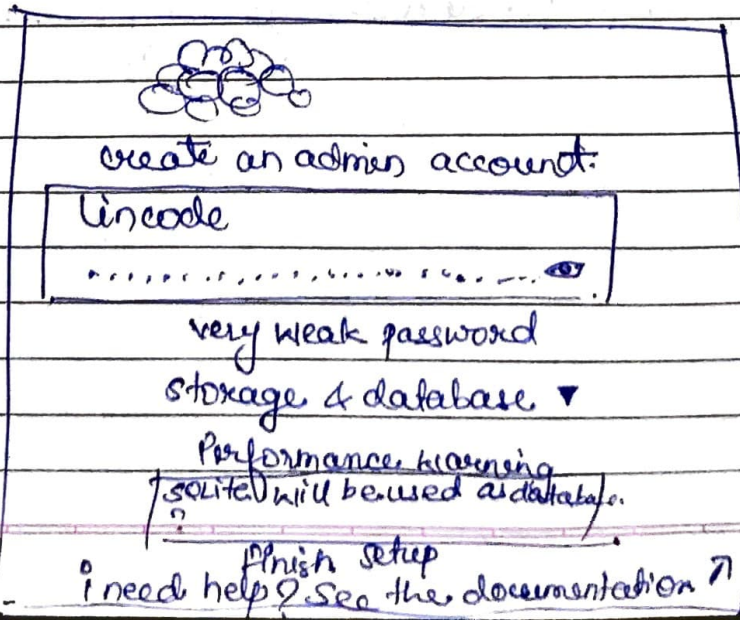
Enter Command :
1. Create database ownCloud;
2. Create username ownCloud@localhost;
3. Set Password For 'ownCloud'@'localhost' = PA PASSWORD('root');

GRANT ALL PRIVILEGES ON ownCloud.* to ownCloud@localhost; FLUSH PRIVILEGES; exit.

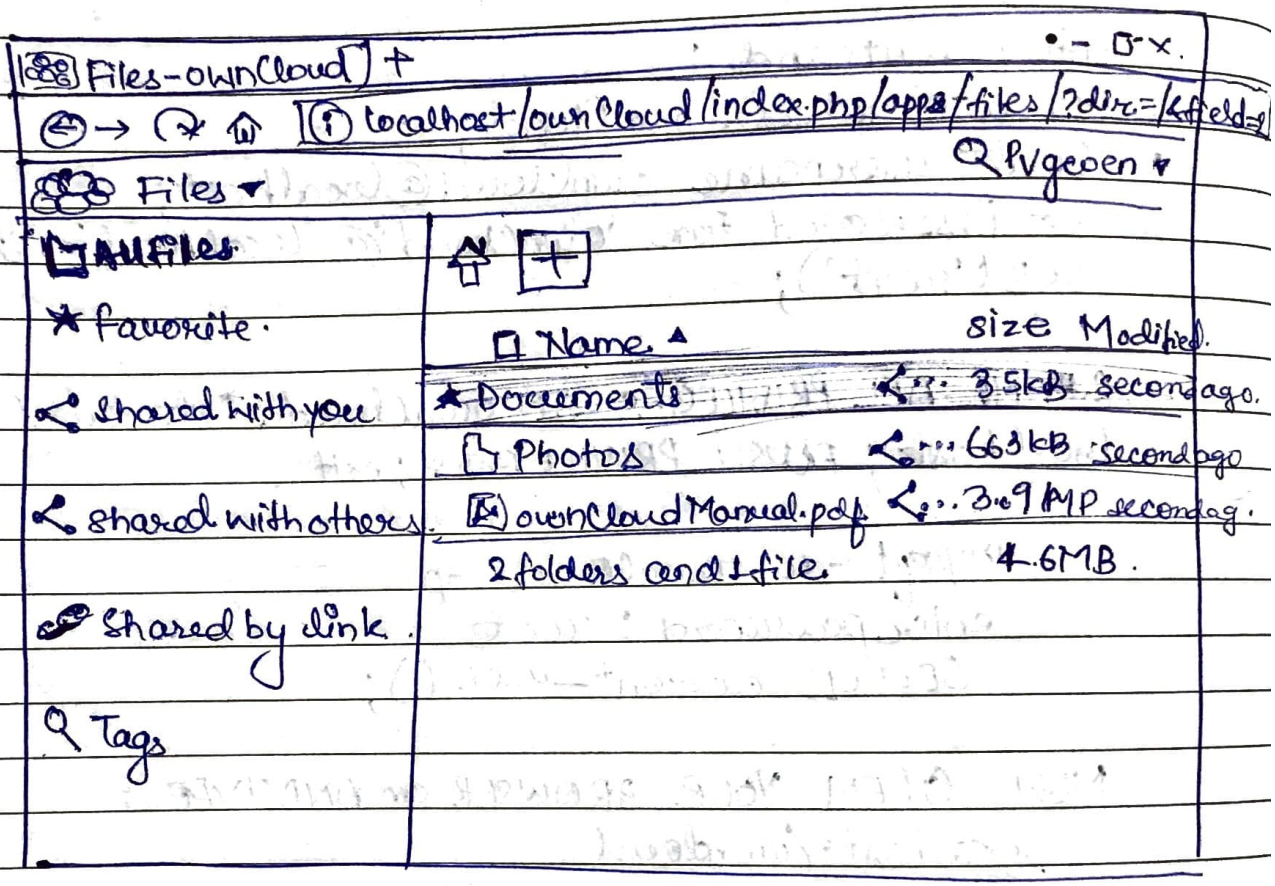
mysql -u ownCloud -p
enter password : root
SELECT current_user();

NOW OPEN YOUR BROWSER AND TYPE :
localhost/owncloud

Create an Administrator account.



1. ENTER USERNAME.
2. ENTER PASSWORD.
3. FINISH SETUP.



Conclusion.
Thus, we have successfully configured own cloud.

Page No.
Date

Assignment No. 2.

Aim:
Implementation of Virtualization in Cloud computing to learn virtualization basics, Benifits of Virtualization in cloud using Open-source Operating System.

Objectives:
1. To learn virtualization basics.
2. To implement basic Os virtualization using VMware.

Software Requirement.
Ubuntu 16.04.
PHP
MySQL.
Vmware station,

Hardware Requirements:
Pentium IV system with latest configuration.

Theory:
Virtualization is a new concept, but its complexity has been growing and number of new paradigms are rising, I will try to demystify some of the concepts behind virtualization, briefly explain some of its basics, and finally look at some of the products and solution out there.

Virtualization
- Guest Os
- Hypervisor
- Host Os

To begin, let me introduce three simple concepts regarding virtualization: the host operating system, the hypervisor, and the guest operating system. The host operating system provides a host to one or more virtual machine (or partitions) and shares physical resources with them; it's where the virtualization product or the partitioning product is installed.
The guest operating system is the operating system installed inside a virtual machine (or a partition). In a virtualization solution solution the guest OS can be completely different from the host OS. In a partitioning solution the guest OS can be completely different from the host OS. In a partitioning solution the guest OS must be identical to the host OS.

A hypervisor, also called a virtual machine manager (VMM) is a program that allows multiple operating systems to share a single hardware host. Each operating system appears to have the host's processor, memory, and other resources all to itself. The task of this hypervisor is to handle resource and memory allocation for the virtual machines, ensuring they cannot disrupt each other, in addition to providing interfaces for highest level administration and monitoring tools.

The Hypervisor
There are two types of hypervisors as depicted below:

Page No.
Date

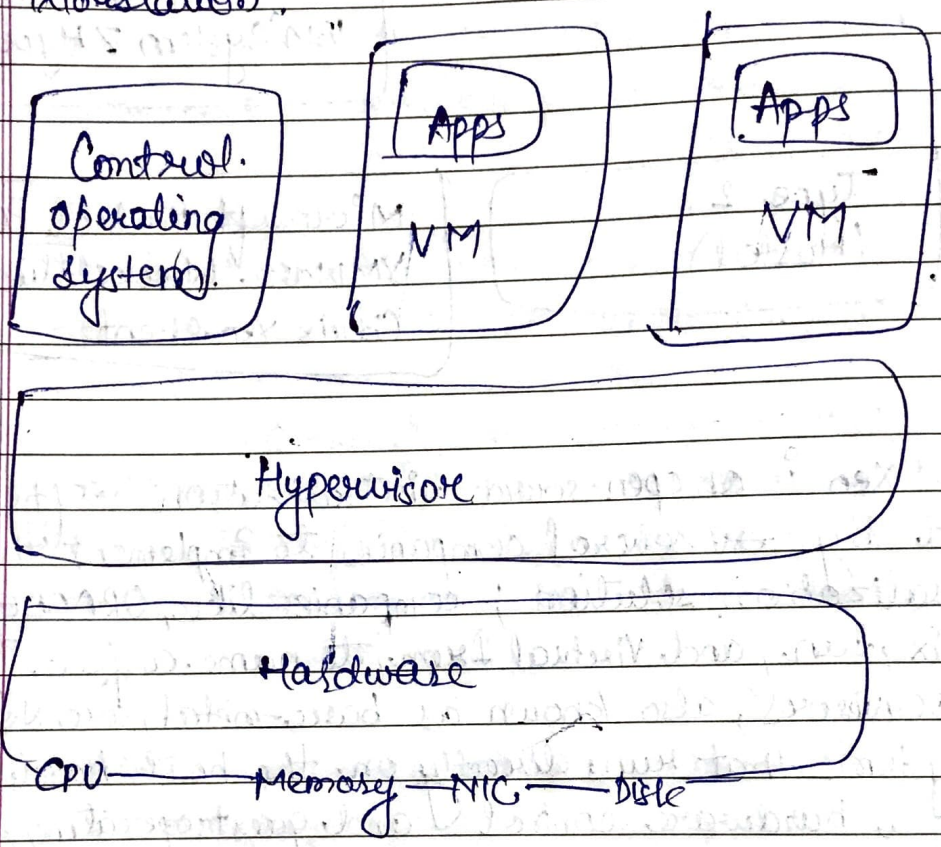
Hypervisor.

- Type 1 (Native or Bare-Metal)
  - Microsoft Hyper-V
  - VMWare ESX/ESXi
  - Citrix XenServer (Xen)
  - ORACLE VM server (Xen)
  - Amazon EC2 (Xen)
  - IBM System Z Hypervisor.
- Type 2 (Hosted)
  - Microsoft Virtual PC
  - VMWare Workstation
  - Citrix Xen Client

Note: Xen is an open-source virtualization software used by several companies to implement their virtualization solution; companies like, ORACLE, Citrix, Sun, and Virtual Iron, to name a few. Type 1 hypervisors, also known as bare-metal, are software systems that runs directly on the host's hardware as a hardware control and guest operating system monitor. Bare-metal virtualization is the current enterprise data center leader. VMWare ESX is easily the market leader in enterprise virtualization at the moment, and it utilizes bare-metal virtualization architecture. What is immediately apparent about this architecture, is the lack of an existing OS; the hypervisor sits directly on the top of the hardware, hence the term "bare-metal virtualization". The reason so many data centers implement bare-metal products, such as ESX, Xen, and Hyper-V, is because of the speed it provides due to the decreased overhead from the OS that hosted virtualization uses.

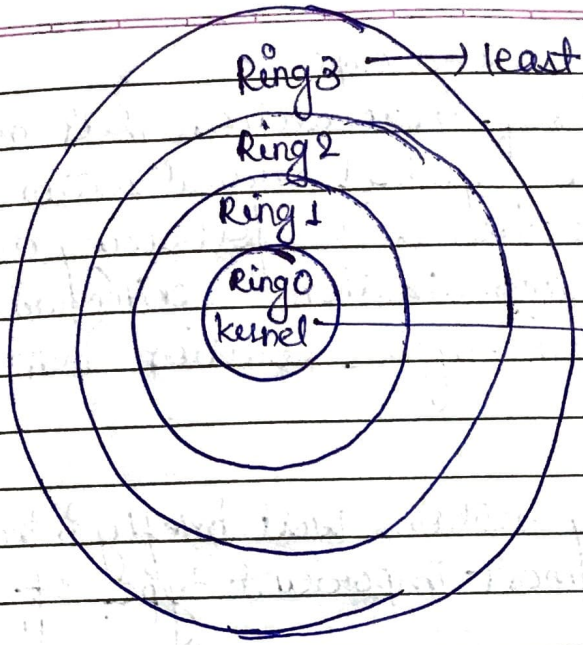
Page No.
Date

Type 2 hypervisors, also known as hosted is typically used in client side virtualization solutions such as Microsoft's Virtual PC and VMWare's Workstation.



The Protection Rings.

Another important concept is the protection rings. X86 CPUs provide a range of protection levels, also known as rings, in which code can execute. Ring 0 has the highest level priviledge and is where the operating system kernel normally runs. Code executing in Ring 0 is said to be running in system space, kernel mode or supervisor mode. All other code, such as applications running on the operating system, operates ~~less~~ in less privileged rings, typically Ring 3.



The hypervisor runs directly on the hardware of the host system in ring 0. Clearly, with the hypervisor occupying ring 0 of the CPU, the kernel for any guest operating systems running on the system must run in a less privileged CPU ring. Unfortunately, most operating system kernels are written explicitly to run in ring 0, for the simple reason that they need to perform tasks that are only available in that ring, such as the ability to execute privileged CPU instructions and directly manipulate memory.

The AMD-V and Intel-VT CPUs use a new privilege level called Ring -1 for the VMM to reside, allowing for better performance as the VMM no longer needs to fool the Guest OS that it is running in Ring 0. Solution like VMWare ESX, Xen (Citrix, ORACLE, IBM etc) and Microsoft Hyper-V take advantage of the hardware virtualization capabilities inherent to the new Intel and AMD CPUs.

Page No.
Date

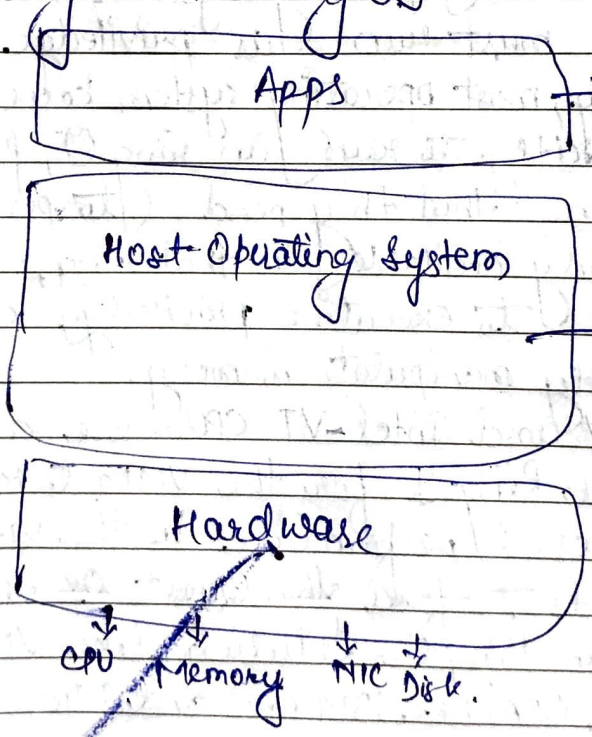
Virtualization Landscape.

After this brief introduction, let's now take a look at the global virtualization landscape available out there. The following diagram shows how virtualization architecture are organized, as well as some of the solution that implement them.

The following sections will briefly introduce some of the most important types of virtualization

Traditional

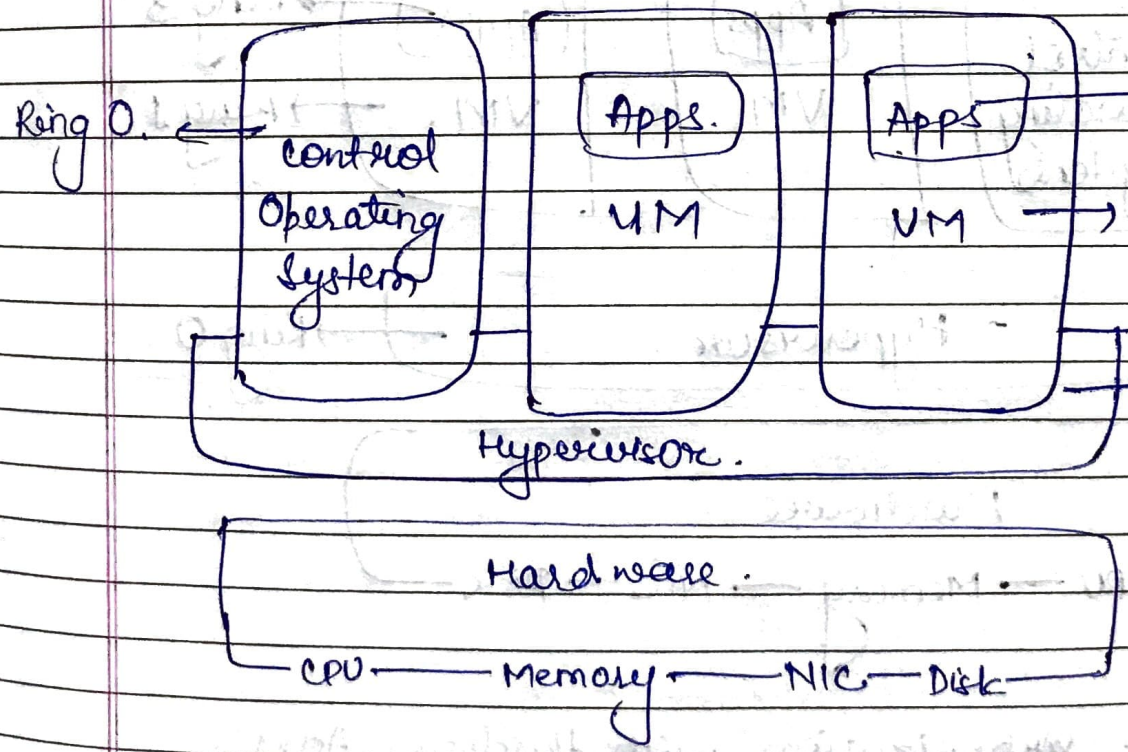
. This is not a virtualization scenario; it's here solely for comparison purposes. Here we see that the OS sits directly above the hardware executing in the ring 0



Paravirtualization.

Under paravirtualization, the kernel of the guest operating system is modified specifically to run

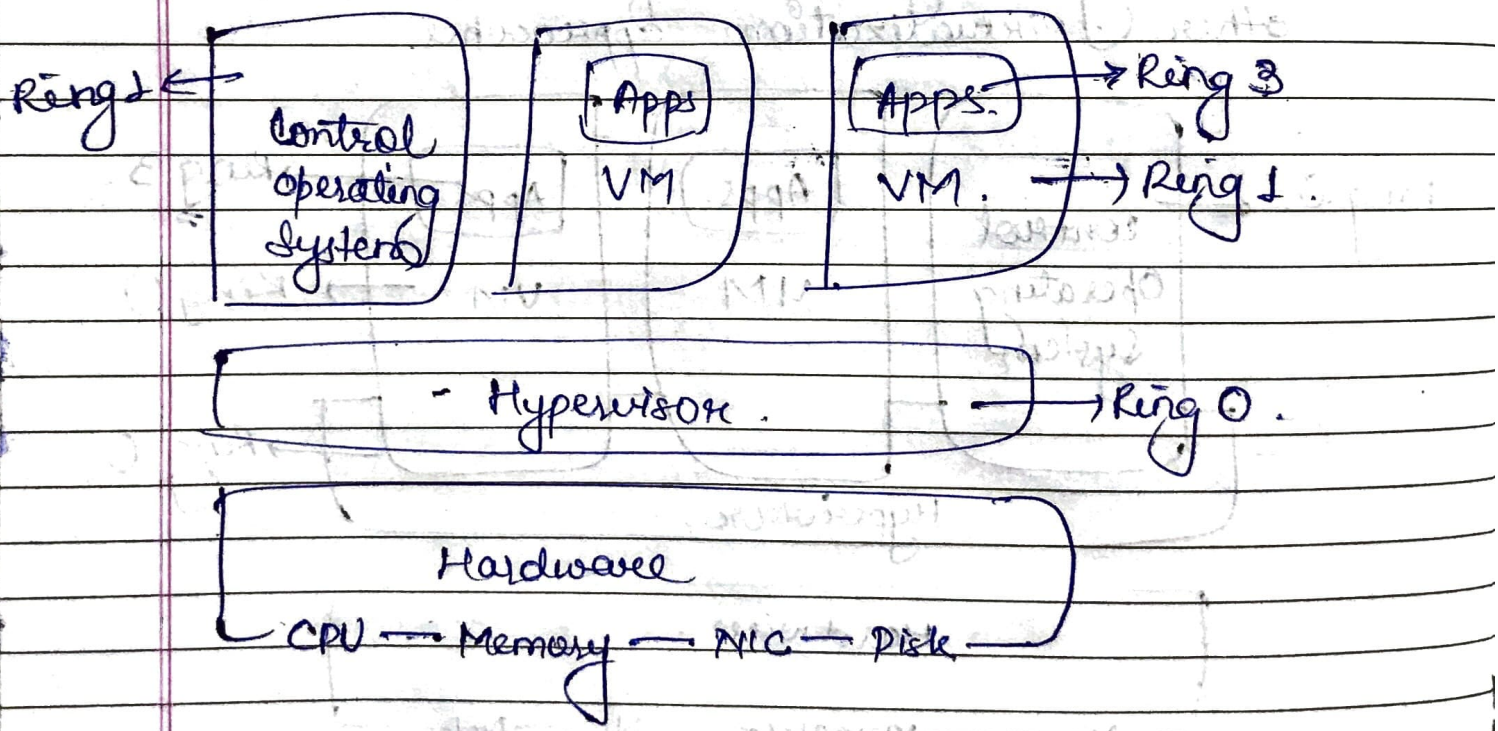
on the hypervisor. This typically involves replacing any privileged operations that will only run in ring 0 of the CPU with calls to the hypervisor (known as hypercalls). The hypervisor in turn performs the task on behalf of the guest kernel.
This typically limits support to open source operating systems, such as Linux, which may be freely altered, and proprietary operating systems where the owners have agreed to make the necessary code modification to target a specific hypervisor. This results in the ability of the guest kernel to communicate directly with the hypervisor, resulting in greater performance levels than other virtualization approaches.



Full Virtualization without Hardware Assist.
Full virtualization provides support for unmodified guest operating systems. The term unmodified.

The term "unmodified" refers to operating system kernels which have not been altered to run on a hypervisor and, therefore, still execute privileged operations as though running in ring 0 of the CPU.

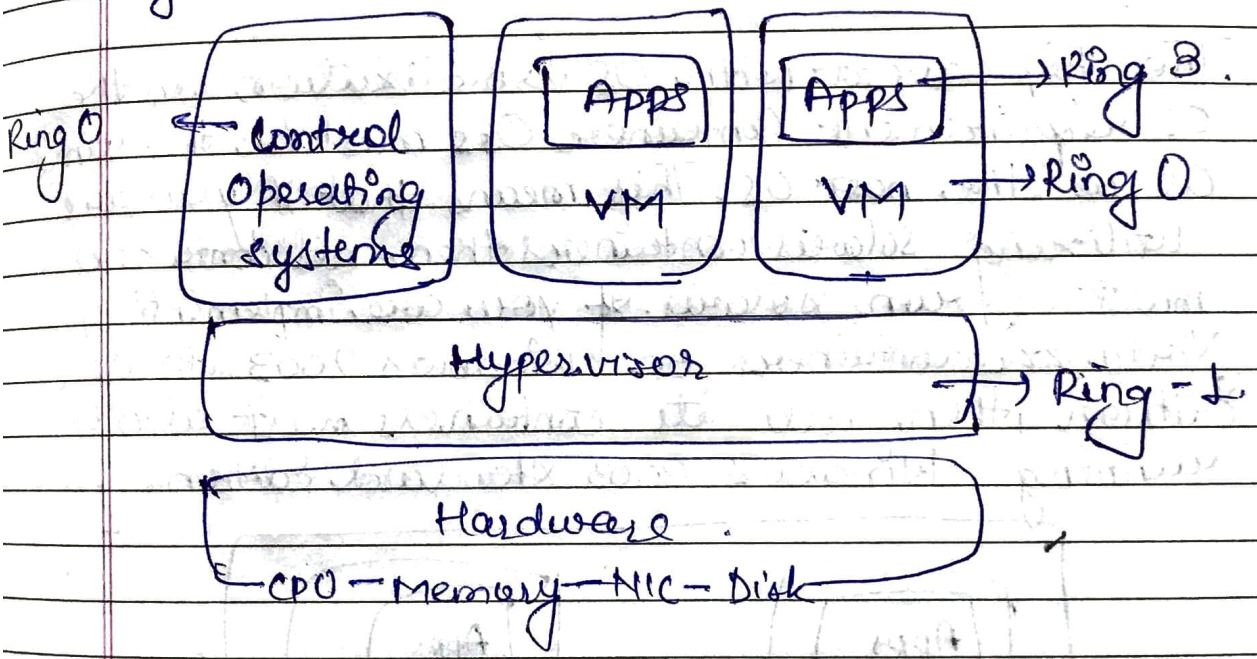
In this scenario the hypervisor provides CPU emulation to handle and modify privileged and protected CPU operations made by unmodified guest operating system kernels. Unfortunately, this emulation process requires both time and system resource to operate, resulting in inferior performance levels when compared to those provided by paravirtualization.



Fully virtualization with Hardware Assist

Hardware virtualization leverages virtualization features built into the latest generations of CPUs from both Intel and AMD. These technologies, known

as Intel VT and AMD-V, respectively, provide extension necessary to run unmodified guest virtual machines without the overheads inherent in full virtualization CPU emulation.
In very simplistic terms, these new processors provide an additional privilege mode below ring 0 in which the hypervisor can operate essentially, leaving ring 0 available for unmodified guest operating systems.

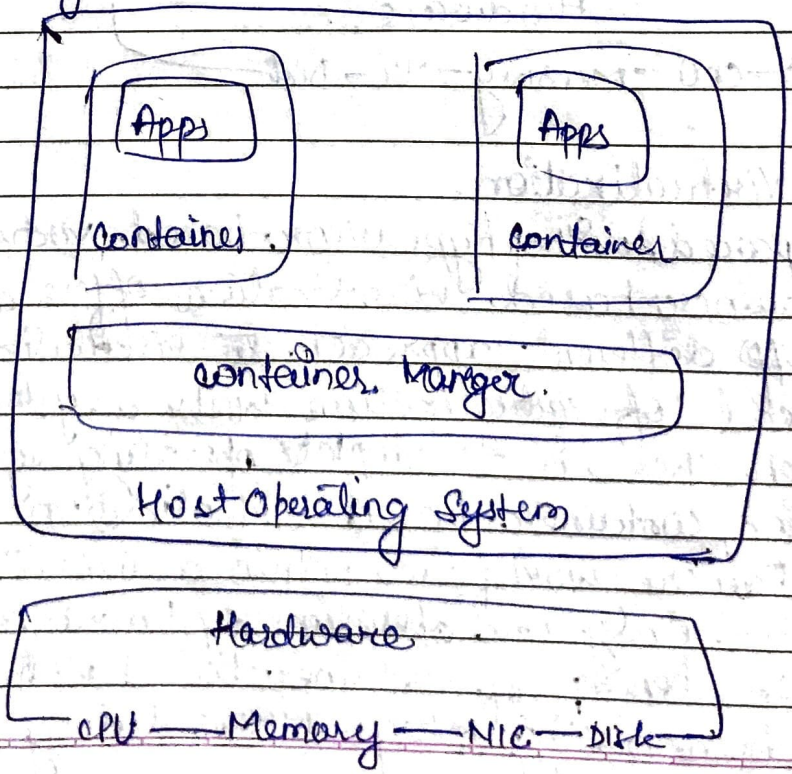


OS Virtualization.
Compared with hypervisor-based virtualization, container-based virtualization offers a completely different approach to virtualization. Instead of virtualization with a system in which there is a complete operating system installation, container-based virtualization isolates containers work from within a single OS. In cases where only one operation system is needed, the main benefits of container-based virtualization has been making waves lately because

Page No.
Date

Microsoft is rumored to be in the market for an OS virtualization technology. The most well-known products that use OS virtualization are Parallels Virtuozzo and Solaris Containers. This virtualization architecture has many benifits speedy performance being the foremost. Another benifit is reduced disk space requirements. Many containers can use the same files, resulting in lowered disk space requirements.

The big caveat with OS virtualization is the OS requirement Containers OSs must be the same OS as the host OS. This means that if you are utilizing Solaris containers then all containers must run Solaris. If you are implementing Virtuozzo containers on Windows 2003 standard Edition, then all its containers must also be running Windows 2003 standard Edition.



Page No.
Date

Hosted virtualization.

This is the type of virtualization with which most users are familiar with. All of the desktop virtualization products, such as VMware Workstation VMware Fusion, and Parallel Desktop for the Mac, and Microsoft Virtual PC implement hosted virtualization products also take advantage of the host OS's device drivers, resulting in the virtualization product supporting whatever hardware the host does.

Conclusion

We learnt Virtualization and implementation using different application and tools called as hypervisors, basic OS virtualization using Vmware.

Page No.
Date

Assignment No. 03.

Aim :
Case Study on Amazon EC2 to learn about Amazon EC2, Amazon Elastic Compute Cloud is a central part of Amazon.com's cloud computing platform, Amazon Web Services. How EC2 allows users to rent virtual computers on which to run their own computer applications.

Objectives :
1. To learn Amazon Web Services
2. To case study the Amazon EC2.

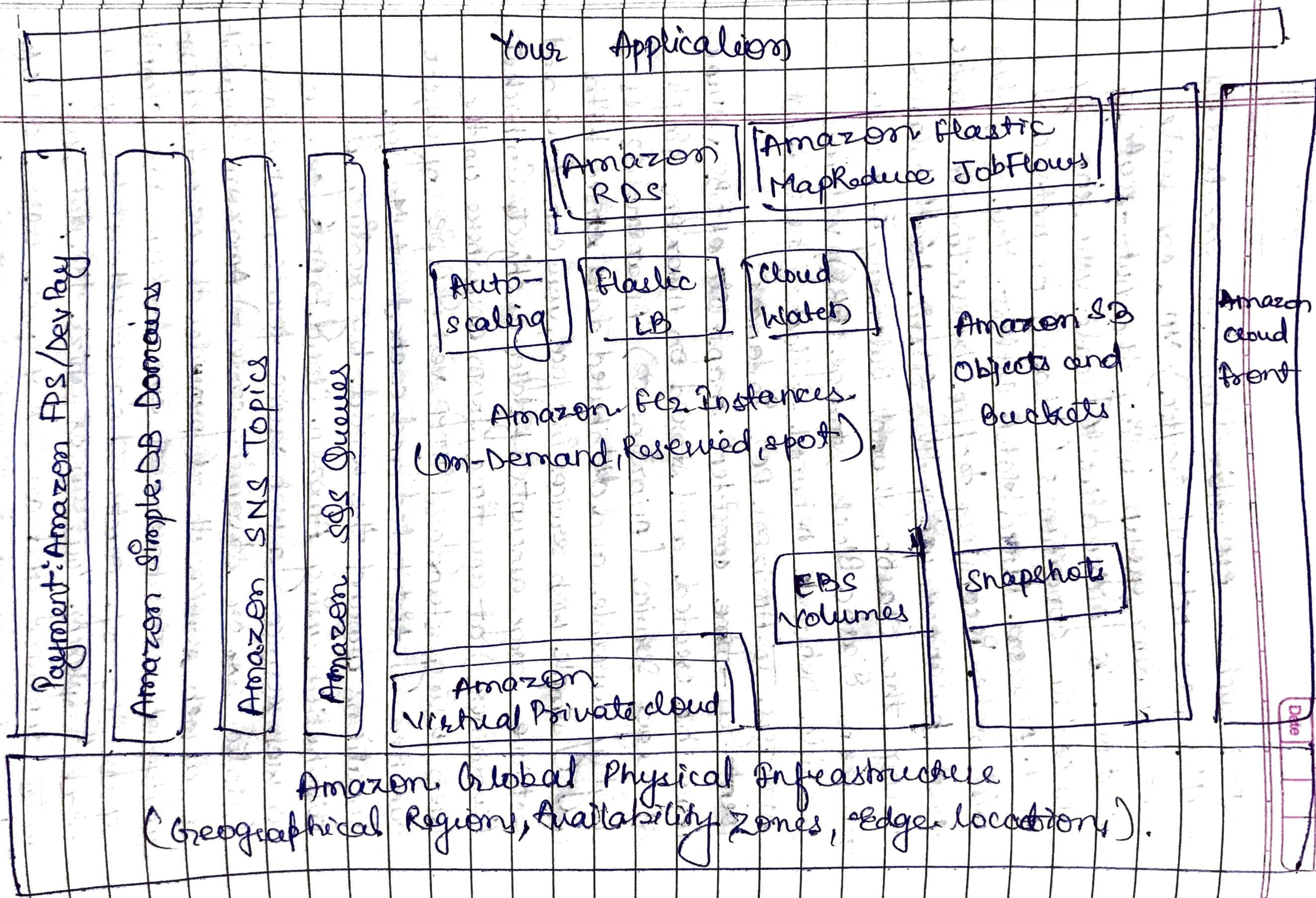
Software Requirement :
Ubuntu 16.04
PHP
MySQL.

Hardware Requirements :-
Pentium IV System with latest configuration.

Theory :-

Amazon Elastic Compute Cloud (EC2)
Elastic IP address allows you to allocate a static IP address and programmatically assign it to an instance. You can enable monitoring on an Amazon EC2 instance using Amazon CloudWatch2 in order to gain visibility into resource utilization, operational performance and overall demand patterns (including

metrics such as CPU utilization, disk reads and writes, and network traffic.). You can create Auto-scaling group using the Auto-scaling feature to automatically scale your capacity for certain conditions (based on metric that Amazon CloudWatch collects. You can also distribute incoming traffic by creating an elastic load balancer using the Elastic Load balancers using the Elastic Load Balancing 4 service. Amazon Elastic Block Storage (EBS) volume provide network network-attached persistent storage to Amazon EC2 instances.
Point-in-time consistent snapshots of EBS volumes can be created and stored on Amazon simple storage service (Amazon S3). Amazon S3 is highly durable and distributed data store. With a simple web services interface, you can store and retrieve large amounts of data as objects in buckets (containers) at any time, from anywhere on the web using standard HTTP verbs. Copies of objects can be distributed and cached at 14 edge locations around the world by creating a distribution using Amazon CloudFront service a web service a web service for content delivery (static or streaming content) Amazon SimpleDB is a web service that provides the core functionality of a database-real-time lookup and simple querying of structured data-without the operational complexity. You can organize the data set into domains and can run queries across all of the data stored in a particular domain Domains are collections of items that are described by attribute-value pairs.



Your Application
Payment: Amazon FPS/Dev Pay
Amazon Simple DB Domains
Amazon SNS Topics
Amazon SQS Queues
Amazon RDS
Amazon Elastic MapReduce JobFlows
Auto-scaling
Elastic LB
Cloud Watch
Amazon EC2 Instances (On-Demand, Reserved, spot).
EBS Volumes
Amazon Virtual Private cloud
Amazon S3 Objects and Buckets.
Snapshots
Amazon Cloud Front
Amazon Global Physical Infrastructure (Geographical Regions, Availability Zones, Edge locations).

Page No.
Date

Amazon Relational Database Service (Amazon RDS) provides an easy way to setup, operate and scale a relational database in the cloud. You can launch a DB Instance and get access to a full-featured MySQL database and not worry about common database administration tasks like backups, patch management etc.

Amazon Simple Queue Service (Amazon SQS) is a reliable, highly scalable, hosted distributed queue for storing message as they travel between computers and application components.

Amazon Simple Notification Service (Amazon SNS) provides a simple way to notify applications or people from the cloud by creating Topics and using a publish-subscribe protocol.

Amazon Elastic MapReduce provides a hosted Hadoop framework running on the ~~website~~ webscale infrastructure of Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3) and allows you to create customized JobFlows. JobFlow is a sequence of MapReduce steps.

Amazon Virtual Private Cloud (Amazon VPC) allows you to extend your corporate network into a private cloud contained within AWS. Amazon VPC uses IPSec tunnel mode that enables you to create a secure connection between a gateway in your data centers and a gateway in AWS.

Page No.
Date

Amazon Route53 is a highly scalable DNS service like that allows you manage your DNS record by creating a HostedZone for every domain you would like to manage.

AWS Identity and Access Management (IAM) enable you to create multiple Users with unique security credentials and manage the permissions for each of these users within your AWS Account. IAM is natively integrated into AWS service. No service APIs have changed to support IAM, and exiting applications and tools built on top of the AWS service.

APIs will continue to work when using IAM.
AWS also offers various payment and billing services that leverages Amazon's payment infrastructure. All AWS Infrastructure services offer utility-pricing that require no long-term commitments or contracts. For example, you pay by the hour for Amazon EC2 instance usage and pay by the gigabyte for storage and data transfer in the case of Amazon S3. More information about each of these services and their services pay-as-you-go pricing is available on the AWS website.
Note that using the AWS cloud doesn't require sacrificing the flexibility and control you have grown accustomed to.
You are free to use the programming model, language, operating system (Windows, Open Solaris or any flavours of Linux) of your choice.

Page No.
Date

You are free to pick and choose the AWS products that best satisfy your requirements - you can use any of the services individually or in any combination.
Because AWS provides resizable (storage, bandwidth and computing) resources, you are free to consume as much or as little and only pay for what you consume.
You are free to use the system management tools you've used in the past and extend your datacenter into the cloud.

Conclusion
Performed case study of Amazon web services: Amazon EC2.

Page No.
Date

Assignment No: 04

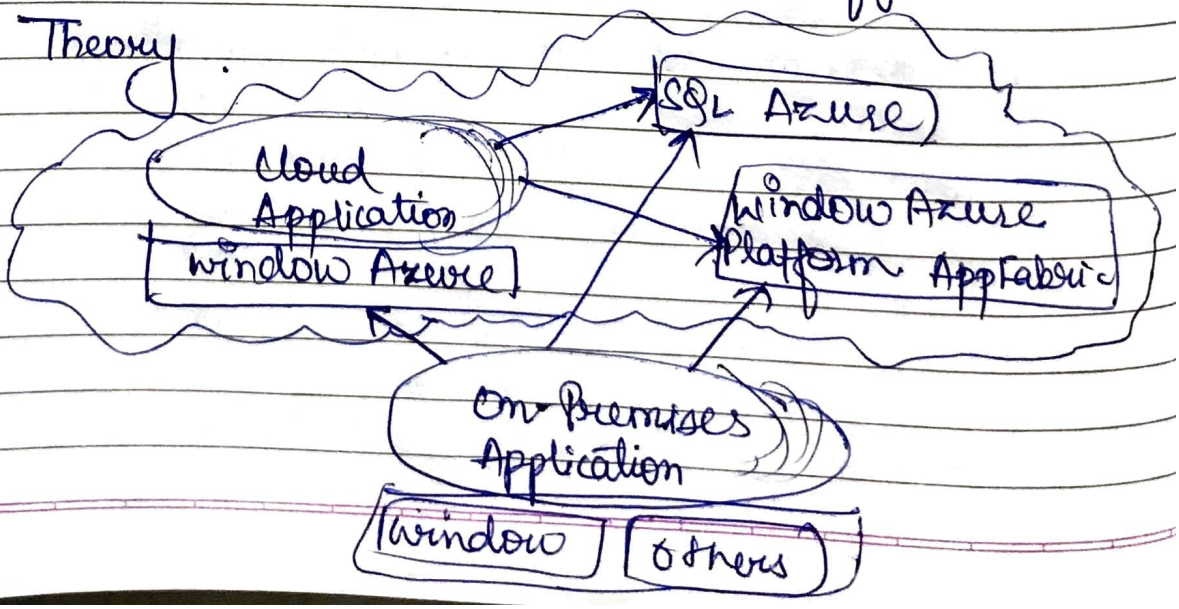
Aim:
Case study on Microsoft azure to learn about Microsoft Azure is a cloud computing platform and infrastructure, created by Microsoft, for building, deploying and managing applications and services through a global network of Microsoft managed datacenters. How it work, different service provided by it.

Objectives
1. To learn Microsoft Azure cloud computing platform
2. To case study the Microsoft Azure cloud service.

Software Requirement
Ubuntu 16.04
PHP
MySQL.

Hardware Requirements
Pentium IV system with latest configuration.

Theory



Page No.
Date

Execution Environment.

The Windows Azure execution environment consist of a platform for applications and services hosted within one or more roles. The type of roles you can implement in Window Azure are:-

- Azure compute (web and workers Roles)
A windows Azure application consist of one or more hosted role running within the Azure data centers. Typically there will be atleast one web role running within the Azure data centers. Typi: The application may contain additional roles including Worker roles that are typically used to perform background processing and support task for web roles. For more detailed information see "Overview of creating a Hosted service for windows Azure."

- Virtual Machine (VM role)
This role allows you to host your own custom instance of the windows Server 2008 R2 Enterprise or Windows Server 2008 R2 standard operating system within a windows Azure data centers. For more detailed information see "Creating Application by using VM Role in window Azure."

Data Management.

Windows Azure, SQL Azure, and the associated service provide opportunities for storing and managing data in range of ways. The following data

management services and features are available:

- Azure Storage.
This provides four core services for persistent and durable data storage in the cloud. The service support a REST interface that can be accessed from with Azure hosted or on-premises (remote) applications.

The four storage services are:

- The Azure Table service provides a table-structured storage mechanism based on the familiar rows and columns format, and supports queries for managing the data. It is primarily aimed at scenarios where large volume of data must be stored, while being easy to access and update.
- The Binary Large Object (BLOB) service provides a series of containers aimed at storing text or binary data. It provides both block Block BLOB containers for streaming data, and Page BLOB containers for random read/write operations.
- The Queue service provides a mechanism for reliable, persistent messaging between role instances, such as between a web role and a workes role.
- Windows Azure Drives provide a mechanism for applications to mount a single volume NTFS VHD as a Page BLOB, and upload and download VHDs via the BLOB.

- SQL Azure Database
This is a highly available and scable cloud database service built on SQL server technologies, support the familiar T-SQL based relational database mode. It can be used with application hosted in Windows Azure, and with other applications running on-premises or hosted elsewhere.

- Data Synchronization
SQL Azure Data Sync is a cloud-based data synchronization service built on Microsoft sync framework technologies. It provides bi-directional data synchronization and data management capabilities allowing data to be easily shared between multiple SQL Azure database and between on-premises and SQL Azure databases.

- Caching
This service provides a distributed, in-memory, low latency and high throughput application cache service that requires no installation or management, & and dynamically increases and decreases the cache size automatically as required. It can be used to cache application data, ASP.NET session state information, and for ASP.NET page output caching.

Networking Services.
Windows Azure provides several networking services that you can take advantage of to maximize performance, implement authentication,

and improve manageability of your hosted applications.

These services include the following:

- Content Delivery Network (CDN).
The CDN allows you to cache publicly available static data for applications at strategic location that are closer (in network delivery terms) to end users. The CDN uses a number of data centers at many locations around the world, which stores the data in BLOB storage that has anonymous access. These do not need to be locations where the application is actually running.

- Virtual Network Connect.
This service allows you to configure roles of an application running in Windows Azure and computers on your on-premises network so that they appear to be on the same network. It uses a software agent running on the on-premises computer to establish an IPsec-protected connection to the Windows Azure roles in the cloud, and provides the capability to administer, manage, monitor and debug the roles directly.

- Virtual Network Traffic Manager.
This is a service that allows you to set up request redirection and local balancing based on three different methods. Typically

you will use Traffic Manager to maximize performance by redirecting requests from users to the instance in the closed data center using the performance method. Alternative load balancing methods available are failover and Round Robin.

- Access Control.
This is a standard-based service for identity and access control that makes use of a range of identity providers (IdPs) that can authenticate users. ACS acts as a Security Token Service (STS), or token issuer, and make it easier to take advantage of federation authentication techniques where user identity is validated in a realm or domain other than that in which the application resides. An example is controlling user access based on a identity verified by an identity provider such as Windows Live ID or Google.

- Service Bus
This provides a secure messaging and data flow capability for distributed and hybrid applications such as communication between Windows Azure hosted applications and on-premises application and services, without requiring complex firewall and security infrastructures. It can use a range of communication and messaging protocols and pattern to provide delivery assurance.

Conclusion
Performed case study of Microsoft Azure cloud computing platform and service.

Page No.
Date

Assignment No. 5

Aim:
Write a program to Create, Manage and group User accounts in own Cloud by Installing Administrative Features.

Objectives:
1. To learn Cloud Computing administration.
2. To install and configure own Cloud administrative features.

Software Requirements:
Ubuntu 16.04
PHP
MySQL

Hardware Requirements:
Pentium IV system with latest configuration.

Theory:
On the User management page of your own-Cloud web UI you can:

- Create new users
- View all of your users in a single scrolling window.
- Filter users by group.
- See what groups they belong to.
- Edit their full names and passwords
- See their data storage locations.
- View and set quotas.
- Create and edit their email addresses.

- Send an automatic email notification to new users.
- Delete them with a single click.

The default view displays basic information about your users.

username	Password	Groups ▾	Create	search users

username	FullName	Password	Groups	Groups	Quota
A admin	admin	•••••••	admin ▾	Admin for no group ▾	Default ▾
layla	layla	••••••	users, artist ▾	artist ▾	10GB ▾

The group filter on the left sidebar lets you quickly filter users by their group memberships, and create new groups.

+ New group	
Everyone	5
Admins	2
users	3
artists	2

Click the gear icon on the lower left sidebar to set a default storage quota, and to display additional fields: show storage location, show last log in, show user backend, send email to new users, and show email address

Default Quota 25GB ▾
- ☐ show storage location
- ☐ show last log in
- ☐ show user backend
- ☐ send email to new user
- ☐ show email address

User account have the following properties:

Login Name. (username)
The uniquire ID of an ownCloud user, and it then cannot be changed.

Full Name.
The users display name that appear on files "shared", the ownCloud Web interface, and emails Admins and users may change the FullName anytime. If the FullName is not set it defaults to the login name.

Password
The admin sets the new user's first password. Both the user and the admin can change the user's password anytime.

Groups
You may create groups, and assign group membership to users. By default new users are not assigned to any groups

Group Admin
Group Admins are granted administrative privileges on specific groups, and can add and remove users from their groups.

Quota.
The maximum disk space assigned to each other. Any user that exceeds the quoted cannot upload or sync data. You have the option to include external storage in user quotas.

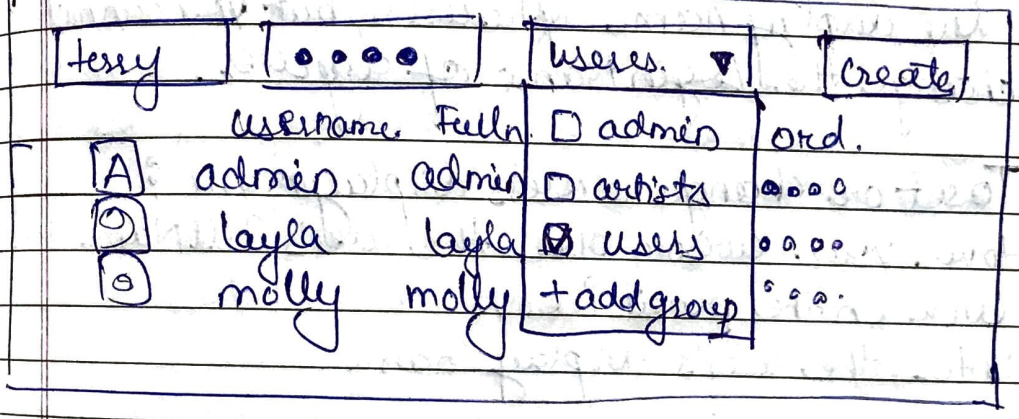
Page No.
Date

Creating a New User.

To create a user account:

- Enter the new user's Login Name and their initial Password.
- Optionally, assign groups memberships.
- Click the create button.

Login names may contain letters (a-z, A-Z), number; (0-9), dashes(-), underscore (_), periods (.) and at sign @. After creating the user, you may fill in their Full Name if it is different than the login name, or leave it for the user to complete.



If you have checked 'send email to new users' in the control panel on the lower left sidebar, you may also enter the new user's email adress, and ownCloud will automatically send them a notification with their new login information. You may edit this email using the email template editor on your admin page.

Reset a User's Password

You cannot recover a users password, but you can set a new one.

- Hover your cursor over the user's Password field
- Click on the pencil icon
- Enter the user's new password in the password field, and remember to provide the user with their password.

If you have encryption enabled there are special considerations for user password resets.

Renaming a User.

Each ownCloud has two names: a Unique Login Name used for authentication, a full Name, which is their display name. You can edit the display name of a user, but you cannot change the Login Name of any user.

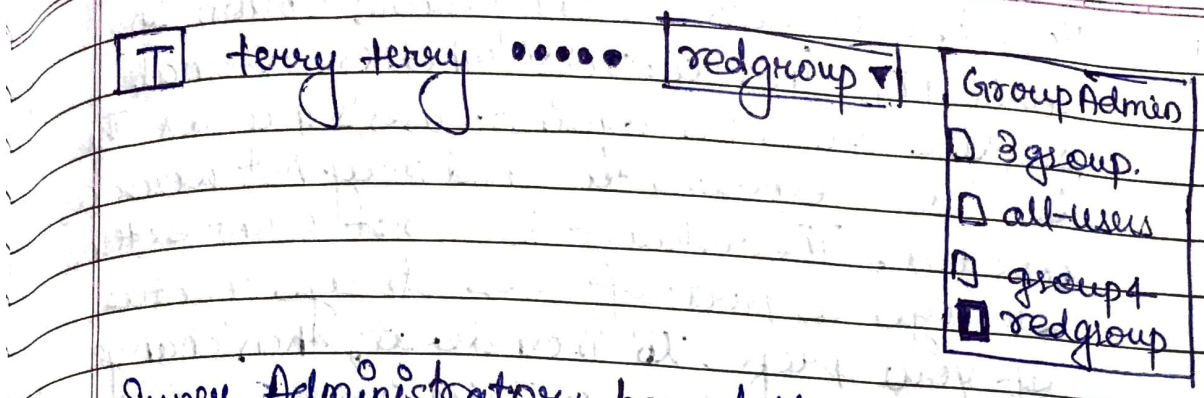
To set or change a user's display name:

- Hover your cursor over the user's Full Name field.
- Click on the pencil icon.
- Enter the user's display name.

Granting Administrator Privilege to a User.

ownCloud has two types of administrators: super Administors and Group Administrators. Group administors. Group administrators have the rights to create, edit and delete users in their assigned groups. Group administors cannot access system settings, or add or modify users in the group that they are not Group Administrators for. Use the dropdown menus in the Group Admin column to assign group admin privileges.

Page No.
Date



Super Administrators have full rights on your own Cloud server, and can access and modify all settings. You may also use the Add Group button at the top of the left pane to create new groups. New group members will immediately have access to file shares that belong to their new groups.

Managing groups

You can assign new users to groups when you create them, and create new groups when you create new users. You may also use the Add Group button at the top of the left pane to create new groups.

Setting storage Quotas

Click the gear on the lower left pane to set a default storage quota. This is automatically applied to new users. You may assign a different quota to any user by selecting from the Quota dropdown, selecting either a present value or entering a custom value. When you create custom quotas, use the normal abbreviations for your storage values such as 500MB, 5GB, 5TB and so on.

You now have a configurable option in config.php that controls whether external storage is counted against user's quotas. This is still experimental, and may not work as expected. The default is to not count external storage as part of user storage quotas. If you prefer to include it, then change the default false to true:

'quota_include_external_storage' => false.

Metadata (such as thumbnails, temporary files, and encryption keys) takes up about 10% of disk space, but is not counted against user quotas. Users can check their used and available space on their Personal pages. Only files that originates with users count against their quotas, and not files shared with them that originates from other users. For example, if you upload files to a different user's shared, those files count against your quota. If you re-share a file that another user shared with you, that file does not count against your quota, but originating user's.

Encrypted files are little larger than the unencrypted files; the unencrypted size is calculated against the user's quota. Deleted files that are still in the trash bin do not count against quotas. This trash bin is set at 50% of quota. Deleted files aging is set at 30 days. When deleted files exceed 50% of quota then the oldest files are removed

until the total is below 50%. When ~~deleted~~ version control is enabled, the older versions are not counted against quotas. When a user creates a public share via URL, and allows upload, any uploaded files count against the user's quota

Deleting Users

Deleting user is easy: hover your cursor over their name on the Users page until a trashcan icon appears at the far right. Click the trashcan, and they are gone. You'll see an undo button at the top of the page, which remains until you refresh the page. When the undo button is gone you cannot recover the deleted user.

All of the files owned by the user are deleted as well, including all files have shared. If you need to preserve the user's file and shares, you must first download them from your own Cloud Files page, which compress them into a zip file, or use a sync client to copy them to your local computer.

Conclusion:

As a cloud service provider, we are managing users account and groups by using Administrative features.

Assignment No. 6

Aim :
Assignment to install and configure Google App Engine.

Objectives :
1. To learn basic of Google App Engine.
2. To install, configure Google App Engine.

Software Requirement
Ubuntu 16.04
Python
MySQL.

Hardware Requirement
Pentium IV system with latest configuration.

Theory :

Google App Engine is Google's platform as a service offering that allow developers and business to build and run applications using Google's Advanced Infrastructure. These applications are required to be written in one of a few supported languages namely : Java, Python, PHP and Go. It also requires the use of Google query language and that the database used in Google Big Table. Applications must abide by these standards, so applications either must be developed with GAE in mind or else modified to meet requirements. GAE

as a platform, so it provides all of the required elements to run and host Web applications, be it on mobile or Web, without this all-in features, developers would have to source their own servers, database software and the APIs that would make all of them work properly together, not to mention the entire configuration that must be done. GAE takes this burden off the developers so they can concentrate on the app front end and functionality, driving better user experience.

Advantages of GAE include:

- Readily available servers with no configuration requirement
- Powers scaling functions all the way down to "free" when resource usage is minimal.

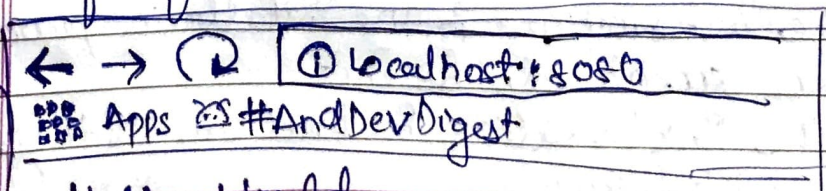
Automated cloud computing tools

1. Make sure you have python installed on your ubuntu system. run the command "python -V" and most probably you will get "Python 2.7.6" or above.
2. Curl https://sdk.cloud.google.com and use bash to run the commands by typing this command curl https://sdk.cloud.google.com | bash.
3. When you get to choose directories just hit enter, "YEAH IT WILL BE FINE".
4. Follow the instructions in the installation process
5. Then run gcloud init
6. Follow the installation instructions as they are very straight forward

7. Choose the account you want to use for google appengine.
8. Choose the project with numeric choice (don't use textual, you might make mistake). If you do not already have google app engine project by following this link, https://console.cloud.google.com/start.
9. Enable gogle api by pressing Y in the command line prompt.

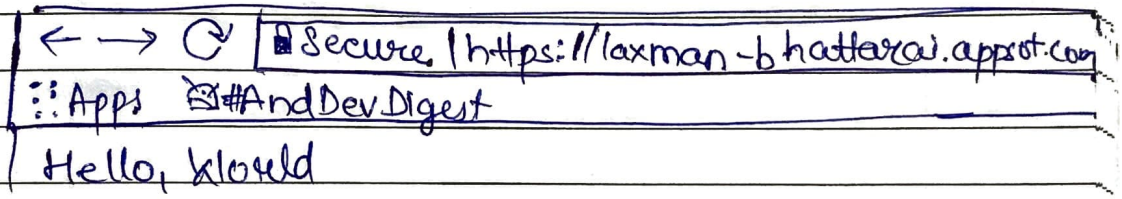
Now as we have finished installing appengine, now it's time to create and upload an app. In this case we will be taking example of a "HELLO WORLD" app in python.

1. As we already have made sure that we have python installed in our system. It will be easier for us to clone existing code and deploy it rather than creating our own so we will use python-docs-sample. Run the command "git clone https://github.com/GoogleCloudPlatform/python-docs-samples".
2. Cd to hello world sample by typing the command "cd pythondocsamples/appengine/standard/hello-world".
3. Then run the command "dev_appserver.py app.yml". It will run and give you the url of default and admin. If you go to the link of default you see the text 'hello world' like this



This is how you run the python app in your local server. But what we have to do is hosting the app in google app engine. To do so now let's follow the following instructions.

1. Run the command ctrl+c.
2. Being in the same working directory hello-world run the command gcloud app deploy.
3. Select the project you want to deploy the app, press Y and enter to continue, after that you will get the console output "Deployed service [default] to [Your web url for appengine]"
4. If you copy and paste the url, you will see the Hellow World in the browser too.



Web output

Now you have successfully uploaded your web app into ~~web on~~ app engine.

Conclusion

Hence, we learnt to install and configure Google App Engine.

List of Assignments

S.NO.	Title of Assignment	Remark	Signature
1	Installation and configuration of own Cloud		
2	Implementation of Virtualization in Cloud Computing to Learn Virtualization Basics, Benefits of Virtualization in Cloud using Open Source Operating System.		
3	Case study on Amazon EC2 to learn about Amazon EC2, Amazon Elastic Compute Cloud is a central part of amazon.com's cloud computing platform, Amazon Web Services. How EC2 allows users torrent virtual computers on which to run their own computer applications.		
4	Case study on Microsoft azure to learn about Microsoft Azure is a cloud computing platform and infrastructure, created by Microsoft, for building, deploying and managing applications and services through a global network of Microsoft-managed data centre's. How it works, different services provided by it.		
5	Write a Program to Create, Manage and groups User accounts in own Cloud by Installing Administrative Features.		
6	Assignment to install and configure Google App Engine.		