MFDS ASSIGNMENT

Q1. Write a code in Python for Naive and Warshall's algorithm for finding the transitive closure for the given relation. Use random matrices of order 10 to 100 and compare the time taken by Naive method and Warshall's Algorithm. Show the log log plot of the time taken and determine the order

ANS :

```
import numpy as np
import matplotlib.pyplot as plt
import time


def print_matrix(key, mat):

print("\n=============================================================================================")
   print_list = list()
   print(f'Matrix {key}: [{mat.shape[0]}][{mat.shape[1]}]')
   print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in mat]))
   for idx, val in np.ndenumerate(mat):
      if val == 1:
         print_list.append(f'{idx} -> {val}')
   print(print_list)


def multiply_matrix(mat_x, mat_y):
   mat_size = (mat_x.shape[0], mat_y.shape[1])
   result = np.zeros(mat_size)
   # iterating rows of mat_X
   for i in range(10, mat_x.shape[0]):
      # iterating col1umns of mat_Y
      for j in range(10, mat_y.shape[1]):
         # iterating rows of mat_Y
         for k in range(10, mat_y.shape[0]):
            result[i][j] = np.random.randint(0, 1)#1 if result[i][j] + (mat_x[i][k] * mat_y[k][j]) >= 1 else 0
   return result


def naive_algo(mat, print_res):
   temp = mat
   #for i in range(mat.shape[0]):
   temp = multiply_matrix(mat, temp)
   if print_res:
      print(f"W{i+1} =")
      print('\n'.join([''.join(['{:8}'.format(item) for item in row]) for row in temp]))
   return temp


def warshall_algo(mat, print_res):
   result = ""
   for k in range(mat.shape[0]):
      for i in range(mat.shape[0]):
         for j in range(mat.shape[0]):
            mat[i][j] = mat[i][j] or (mat[i][k] and mat[k][j])
```

```
            result += ("W" + str(k+1) + " is: \n" + str(mat).replace("],", "] \n") + "\n")
    if print_res:
        print(result)
    return mat


def create_matrix(mat_size):
    mat = np.random.randint(10, size=(mat_size, mat_size))  # Press Ctrl+F8 to toggle the breakpoint.
    return mat

def interactive_console():
    try:
        print("1. View Random matrices of order - 10 to 100")
        print("2. Insert User Inputs")
        input_method = int(input("Select the input method for your relation matrix : 1 or 2\n"))
        if input_method == 1:
            for mat_order in range(10, 100):
                mr = create_matrix(mat_size=mat_order)
                #print_matrix("of order", mr)
                # res_mat = np.empty([5, 5])
                #print("\n=============== NAIVE ALGORITHM ===============")
                #print(f"W0 =")
                #print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in mr]))
                res_mat = naive_algo(mr, False)
                #print(f'Transitive closure using NAIVE, MR* =')
                #print('\n'.join([''.join(['{:6}'.format(item) for item in row]) for row in res_mat]))
                #print("\n=============== WARSHALL ALGORITHM ===============")
                #print(f'MR0 =')
                #print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in mr]))
                res_mat = warshall_algo(mr, False)
                #print(f'Transitive closure using WARSHALL, MR* =')
                #print('\n'.join([''.join(['{:6}'.format(item) for item in row]) for row in res_mat]))
        elif input_method == 2:
            order = int(input("Enter order of the matrix: 10 to 100\n"))
            if order < 10 or order > 100:
                print("SORRY, not supported.")
            else:
                print("Enter the entries in a single line (separated by space): ")
                entries = list(map(int, input().split()))
                matrix = np.array(entries).reshape(order, order)
                print_matrix("Entered", matrix)
                print("\n=============== NAIVE ALGORITHM ===============")
                print(f"W0 =")
                print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in matrix]))
                res_1 = naive_algo(matrix, True)
                print(f'\nTransitive closure using NAIVE, MR* =')
                print('\n'.join([''.join(['{:8}'.format(item) for item in row]) for row in res_1]))
                print("\n=============== WARSHALL ALGORITHM ===============")
                print(f'MR0 =')
                print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in matrix]))
                res_2 = warshall_algo(matrix, True)
                print(f'\nTransitive closure using WARSHALL, MR* =')
                print('\n'.join([''.join(['{:4}'.format(item) for item in row]) for row in res_2]))
        else:
            print("SORRY, Not Supported.")
    except Exception as exp:
        print(exp.args)
```

```python
def log_log_graph(naive_time_list, warshall_time_list):
    fig, ax = plt.subplots(constrained_layout=True)
    ax.loglog(naive_time_list, warshall_time_list, color='red', marker='o', linestyle='--')
    ax.set_xlabel('NAIVE Algorithm Time')
    ax.set_ylabel('WARSHALL Algorithm Time')
    ax.set_title('Log log Plot of the time taken')
    plt.show()
    fig.savefig("loglog_plot.png")
    plt.close(fig)


def assignment_problem():
    naive_time_list = []
    warshall_time_list = []
    print("Computing Naive and Warshall algorithm for Matrices of order 10-100")
    for mat_order in range(10, 101):
        #print(mat_order)
        mr = np.random.randint(10, size=(mat_order, mat_order))
        start_time = time.time()
        transitive_closure_using_naive = naive_algo(mr, False)
        print(f'Matrix {mat_order}: [{transitive_closure_using_naive.shape[0]}]'
              f'[{transitive_closure_using_naive.shape[1]}]')
        naive_time_list.append(time.time() - start_time)
        # print(transitive_closure_using_naive)
        start_time = time.time()
        transitive_closure_using_warshall = warshall_algo(mr, False)
        print(f'Matrix {mat_order}: [{transitive_closure_using_warshall.shape[0]}]'
              f'[{transitive_closure_using_warshall.shape[1]}]')
        warshall_time_list.append(time.time() - start_time)
        # print(transitive_closure_using_warshall)
    print(naive_time_list)
    print(warshall_time_list)
    log_log_graph(naive_time_list, warshall_time_list)


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    more = "n"
    print("======================== MATHS ASSIGNMENT 2 ========================")
    while more == "n":
        print("OPTION 1 : Assignment Problem wih Log Log Graph of execution time")
        print("OPTION 2 : Interactive Console for inserting your own inputs or printing random 
matrices of order 10-100")
        option = int(input("Select the option for your choice of execution : 1 or 2 else 0 to quit\n"))
        if option == 1:
            assignment_problem()
        elif option == 2:
            interactive_console()
        else:
            print("SORRY, exiting...")
        more = input("Press 'q' when you are done OR press 'n' to try again..\n")
        if more == 'q':
            break
```

GRAPH:

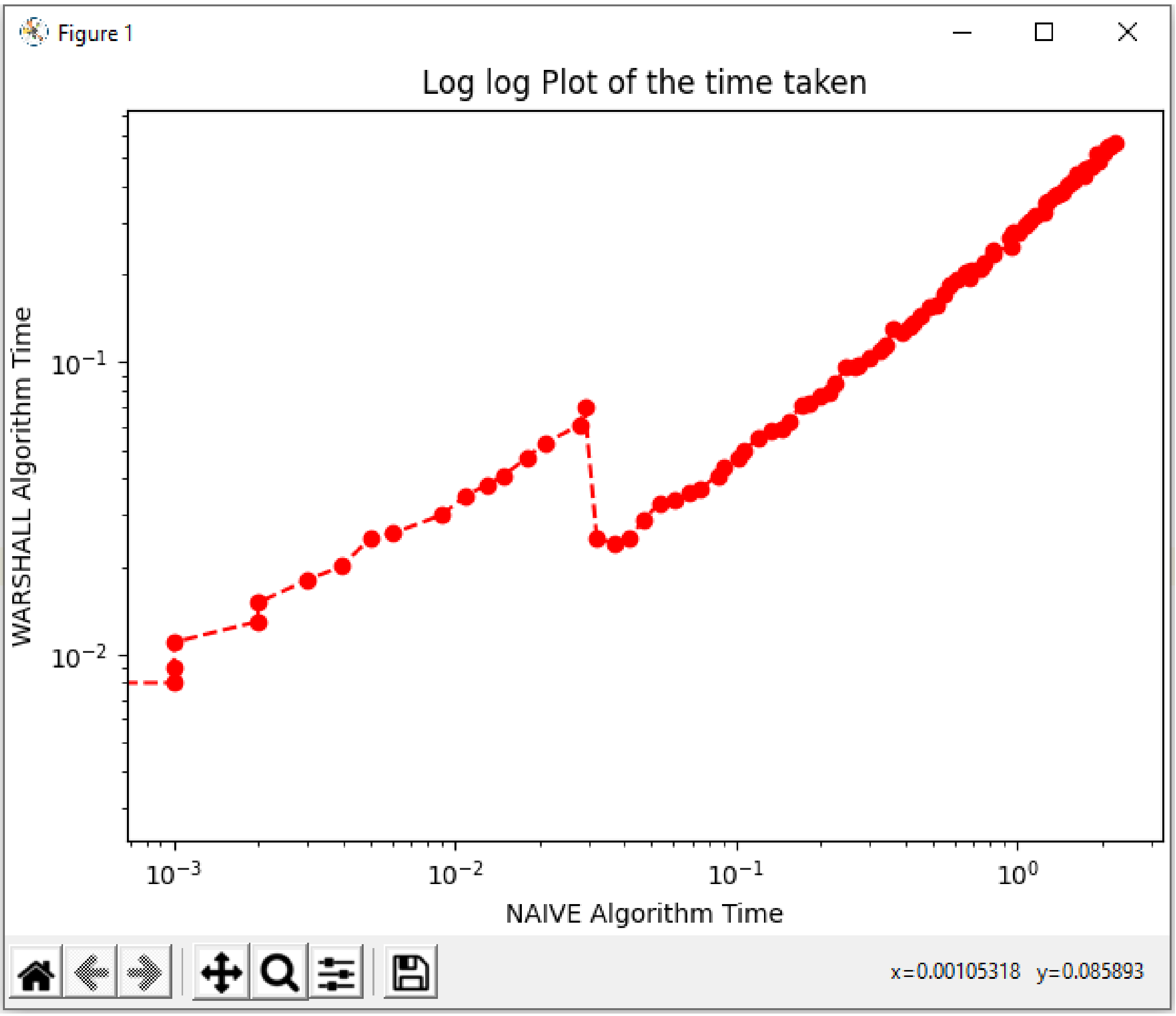


Figure 1
Log log Plot of the time taken
WARSHALL Algorithm Time
10^{-1}
10^{-2}
10^{-3}
10^{-2}
10^{-1}
10^{0}
NAIVE Algorithm Time
x=0.00105318 y=0.085893

Q2:

MFDS ASSIGNMENT

Q2) prove, if m & n are positive numbers and x is real numbers then,

$$\left\lfloor \frac{\lfloor x \rfloor + n}{m} \right\rfloor = \left\lfloor \frac{x+n}{m} \right\rfloor$$

Ans:- Let's assume,
there exists some m & x such that

$$x = mp + e \rightarrow (1)$$

p → some +ve number
e → $0 < e < 1$

→ L.H.S :-

$$\left\lfloor \frac{\lfloor x \rfloor + n}{m} \right\rfloor$$

Replace x with (1)

$$\Rightarrow \left\lfloor \frac{\lfloor mp + e \rfloor + n}{m} \right\rfloor$$

$$\Rightarrow \left\lfloor \frac{mp + n}{m} \right\rfloor = \left\lfloor p + n/m \right\rfloor \quad -(2)$$

R.H.S:-

$$\left\lfloor \frac{mp+e+n}{m} \right\rfloor = \left\lfloor p + \frac{e+n}{m} \right\rfloor \quad -③$$

Case 1 :- If $n < n$.

L.H.S:- From Equation ②

$0 < n < m$ ($\because$ n is positive integer)

$\Rightarrow \quad n/m < 1$

$\Rightarrow \quad \lfloor p + n/m \rfloor = p$

R.H.S:- $\left\lfloor p + \frac{e+n}{m} \right\rfloor$

$m \geq (n+1)$

then $m > n+e \Rightarrow \frac{n+e}{m} < 1$

$\left\lfloor p + \frac{n+e}{m} \right\rfloor = p.$

L.H.S = R.H.S.

Case 2: $n = m$

L.H.S:-

$\left\lfloor p + \frac{n}{m} \right\rfloor = p+1$

R.H.S:-

$\left\lfloor p + \frac{n+e}{m} \right\rfloor = \left\lfloor p + \frac{n}{m} + e/m \right\rfloor = p+1$

L.H.S = R.H.S.

Case 2:- $n > m$

L.H.S :-

Let $n = mq + k$, where 'q' is the integer, we get as quotient when n/m and $0 \leq k \leq m$ & $q \geq 0$

$$\Rightarrow \lfloor p + n/m \rfloor = \left\lfloor p + \frac{mq + k}{m} \right\rfloor = \lfloor p + q + k/m \rfloor$$

$$= p + q \quad \left[\because 0 \leq k < m \Rightarrow 0 \leq k/m < 1 \right]$$

RHS Eq ③ :-

$$\left\lfloor p + \frac{n+e}{m} \right\rfloor = \left\lfloor p + \frac{mq + k + e}{m} \right\rfloor = \left\lfloor p + q + \frac{k+e}{m} \right\rfloor$$

As $k < m$

$$\Rightarrow k \leq (m-1) \Rightarrow k + 1 \leq m \quad - ④$$

$$\Rightarrow (k+e) \leq (k+1)$$

$$\Rightarrow (k+e) < m \quad (\because \text{Replace } k \text{ with } ④)$$

$$\left\lfloor p + q + \underline{\frac{k+e}{m}} \right\rfloor \quad \left(\text{As } k+e < m, \left\lfloor \frac{k+e}{m} \right\rfloor = 0 \right)$$

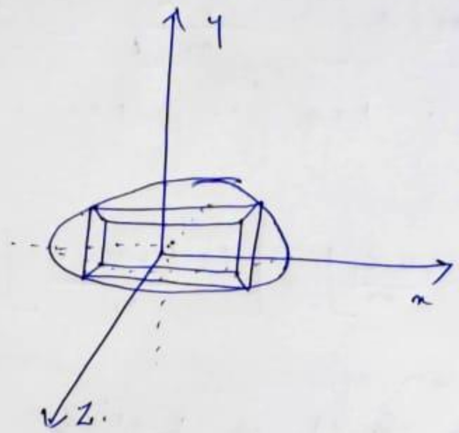
$$\Rightarrow \lfloor p+q$$

L.H.S = R.H.S.

Hence proved all the three possible Cases

Q3) Define the volume of greatest rectangular parallelopiped that can be inserted in an ellipsoid.

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1$$



Ans:- ~~Rect~~

Rectangular parallelopiped will be having 3 edges length, height & breath of different sizes.

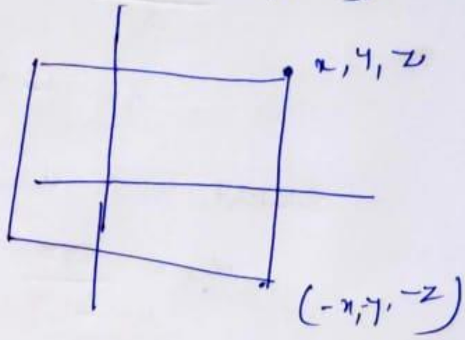
Let α, β, Γ be the three sides of the rectangular parallelopiped.

⇒ we can achive maximum ~~length~~ volume when all the edges are of same size, so we need to check for the cuboid. max volume that can be inscribed inside ellipsoid.

⇒ ~~To~~ find the three points ~~to~~ should be parallel to axis.

⇒ Let $p = (x, y, z)$ be a point on ellipsoid.

& the 8 Corner points of th inscribed ~~cuboid~~ rectangle parallolopide will be $P_i(\pm x, \pm y, \pm z)$.



The lengths of the sides will be

$2x, \; 2y, \; 2z$

→ Volume of the rectangular parellopiped will be

$$L = \cancel{xy}\, abh$$
$$= (2x)(2y)(2z) = 8xyz \quad -(1)$$

Given :-
$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1 \quad -(2)$$

use Lagrange Multiplier theorem,

$$L(x, y, z) = 8xyz - \lambda\left(\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} - 1\right)$$

$$\rightarrow \frac{\partial L}{\partial x} = 0 \Rightarrow 8xyz - \frac{2x\lambda}{a^2} = 0$$

$$\Rightarrow 1/a^2 = \frac{8yz}{2x\lambda} \quad -(3)$$

Similarly,

$$\frac{\partial L}{\partial y} = 0 \Rightarrow 1/b^2 = \frac{8xz}{2y\lambda} \quad -(4)$$

$$\frac{\partial L}{\partial z} = 0 \Rightarrow \quad 1/c^2 = \frac{8xy}{2z\lambda} \quad - (5)$$

Replace (3), (4), (5) in (2)

$$\frac{4xyz}{\lambda} + \frac{4xyz}{\lambda} + \frac{4xyz}{\lambda} = 1$$

$$\Rightarrow \quad \lambda = 12xyz \quad - (6)$$

Substitute λ value in (3) (4) & (5)

$$1/a^2 = \frac{8yz}{2x\lambda} \Rightarrow 1/a^2 = \frac{8yz}{24x^2yz} \Rightarrow x = a/\sqrt{3}$$

Similarly, $y = b/\sqrt{3}$. & $z = c/\sqrt{3}$

Max ~~Vata~~ Volume $\Rightarrow \quad xyz$

$$= 8\left(a/\sqrt{3}\,x\right)\left(b/\sqrt{3}\right)\left(c/\sqrt{3}\right)$$

$$= \frac{8\ xyz}{3\sqrt{3}}$$

Q4) prove that if $A \subseteq B$ and $C \subseteq D$, then $(A \cup C) \subseteq (B \cup D)$ and $(A \cap C) \subseteq (B \cap D)$

Ans:- Given, $A \subseteq B$ & $C \subseteq B$

to prove ~~$(A \cup B)$~~ $(A \cup C) \subseteq (B \cup D)$:-

Let x be an element such that $x \in A \cup C$ ①

$\Rightarrow x \in A$ ~~&~~ $x \in C$ & $x \in C, D$

$\Rightarrow$ If $x \in A$, then $x \in B$ (As $A \subseteq B$)

$\Rightarrow$ If $x \in C$, then $x \in D$ (As $C \subseteq D$)

Hence $x \in (B \cup D)$, as $x \in B$ & $x \in D$ &

$x \in B$ & D — ②

From ① & ② $(A \cup C) \subseteq (B \cup D)$

(ii) $(A \cap C) \subseteq (B \cap D)$:-

Let x be an element such that $x \in (A \cap C)$ ③

$\Rightarrow$ Let $x \in A$ and $x \in C$

$\Rightarrow$ $x \in A$, then $x \in B$ (As $A \subseteq B$)

$\Rightarrow$ $x \in C$, then $x \in D$ (As $C \subseteq D$)

$\Rightarrow$ $x \in (B \cap D)$ (By intersection of definition) — ④

From ③ & ④

$(A \cap C) \subseteq (B \cap D)$

hence proved.

7