In the name of God

استاد : دکتر هراتی

دانشجو : توحید حقیقی سیس

شماره دانشجویی : 830598021

موضوع : تمرین چهارم

تمرین اول :

۱-(پژوهش) تحقیقی از یک منبع معتبر (مجلات پایگاه‌هایی مثل .ACM, IEEE, ScienceDirect, Springer, etc) پیدا کنید که از پیج‌رَنک در کنار کار اصلی خود بهره گرفته باشد. بطور خلاصه در یک صفحه، این کاربرد را توضیح دهید.

Centrality ranking in multiplex networks using topologically biased random walks

Cangfeng Dinga,b

,Kan Lia ,*

مقاله بالا در مورد مرکزیت ها و محاسبه مرکزیت ها به وسیله الگوریتم page rank است در این الگوریتم بر اساس اهمیت پیج ها و hub بودن برخی نود ها .

مشخص کردن مرکزیت آماری قابل توجه گره ها یکی از اهداف اصلی شبکه های مالتی پلکس است. با این حال ، اقدامات مرکزیت فعلی برای رتبه بندی گره ها فقط روی پیاده روی های تصادفی یا ساختار توپولوژیکی شبکه متمرکز هستند .

یک چالش اساسی نحوه سنجش مرکزیت گره ها در شبکه های مالتی پلکس است ، هم به توپولوژی شبکه و هم به انواع مغرضانه پیاده روی های تصادفی ، مانند پیاده روی های مغرضانه که با خواص هر گره به طور جداگانه در هر لایه انجام می شود یا پیاده روی های مغرضانه بستگی دارد. در عوض یک یا حتی بیشتر از ویژگیهای چندگانه ذاتی گره ورود است. در این مقاله ، با در نظر گرفتن این دو جنبه ، ما یک چارچوب ریاضی مبتنی بر پیاده روی تصادفی مغرضانه را پیشنهاد می کنیم ، به نام Multiplex چند جانبه Topologically PageRank ، که به شما امکان می دهد مرکزیت را محاسبه کرده و بر این اساس گره ها را در شبکه های مالتی پلکس رتبه بندی کنید. به طور خاص ، بسته به ماهیت تعصبات و برهم کنش گره ها بین لایه های مختلف ، موارد افزودنی ، ضرب و ترکیبی از Multiplex موضعی را با عنوان PageRank متمایز می کنیم. هر مورد با تنظیم پارامترهای بایاس ، نشان می دهد که چگونه درجه بندی مرکزی گره در یک لایه بر رتبه بندی که ماکت آن در لایه های دیگر به دست می آورد ، تأثیر می گذارد و میزان بازدید واکرها به طور ترجیحی از هاب ها یا گره های ضعیف متصل را ضبط می کند. آزمایشات انجام شده بر روی دو شبکه مالتی پلکس در دنیای واقعی نشان می دهد که مالتی پلکس چند جانبه ای که از نظر توپولوژیکی عمل می کند ، هم از نظر مورد بی طرفانه مربوطه و هم از روش های رتبه بندی فعلی بهتر عمل می کند و می تواند گره هایی را که در شبکه های مالتی پلکس به طور قابل توجهی در بالاترین رتبه قرار دارند .

تمرین دوم :

۲-(طراحی و پیاده سازی الگوریتم)

هدف این تمرین پیاده سازی یک موتور جستجوی ساده با استفاده از دو الگوریتم PageRank و Hub and Authorities است. دیتاستی که در اختیار شما قرار گرفته شامل لیستی از وب‌سایت‌ها، محتوا و موضوع آن است. **برنامه شما باید یک کلمه ورودی از کاربر گرفته و بر اساس رتبه‌بندی انجام شده وب سایت‌های برتر را نمایش لیست کند.**

مجموعه داده به صورت یک فایل JSON است. فرمت هر وب‌سایت در این دیتاست به شکل زیر است:

id: شناسه یکتای وب‌سایت

Content: محتوای وب‌سایت

Links: لیستی از شناسه‌ها که این وب‌سایت به آنها لینک داده است

Category: موضوع وب‌سایت

الف) گراف/شبکه ارجاعات را بسازید و با الگوریتم پیج‌رَنک تحلیل کنید.

ب) گراف/شبکه ارجاعات را بسازید و با الگوریتم پیج‌رَنک موضوعی تحلیل کنید. (در ورودی به جز کلمه کلیدی جستجو، یک موضوع نیز بگیرید)

ج) گراف/شبکه ارجاعات را بسازید و با الگوریتم "Hubs & Authorities" نیز تحلیل کنید.

د) نرم افزار گِفی (Gephi) برای تحلیل گراف/شبکه را نصب کرده و گراف/شبکه ارجاعات را رسم و تحلیل کنید. هر چیزی در مورد این گراف قابل توجه است گزارش کنید.

برای این سوال اول من اومدم از روی فایل های json ماتریس adjency matrix رو تولد کردم و همچین اومدم 3 تا وکتور مربوط به fun و sport و news رو هم ایجاد کردم تا در بخش دوم بتوانم از اون وکتور ها استفاده کنم .

```
import numpy as np
```

```
import json
```

```
f = open('DATASET.json')

# returns JSON object as
# a dictionary
data = json.load(f)

all_ara=get_list(data)

pagerank_array=make_matrix(all_ara,1000)

sport = np.zeros(1001).reshape(1001, -1)
news = np.zeros(1001).reshape(1001, -1)
fun = np.zeros(1001).reshape(1001, -1)

counter=0
for i in data['websites']:
    if(i['category']=="news"):
        news[counter]=1
    if(i['category']=="sport"):
        sport[counter]=1
    if(i['category']=="fun"):
        fun[counter]=1
    counter+=1



np.savetxt("foo.csv", pagerank_array, delimiter=",")
# Closing file
f.close()
```

همان طور که در شکل بالا میبینید از کتابخانه json برای کار با داده جسون استفاده کردم جیسون را خوانده و به یک تابع که در زیر هست پاس دادم تا ان را تبدیل به ارایه 2 بعدی کند و در مراحل بعدی بتوانم از ان استفاده کنم .

```
def get_list(data):
    array_list=[]
    for i in data['websites']:
        array_list.append(i['links'])
    return array_list
```

همان طور که از فایل جسون معلوم است ساختار به صورت درختی در زیر شاخه website و سایت هایی که ان سایت با ان ارتباط دارد در زیر شاخه links قرار دارد .

خروجی این تابع یک ارایه 2 بعدی است .

در مرحله بعدی عکس اول ارایه 2 بعدی را به تابع زیر میدهیم تا ان را تبدیل به ماتریس numpy و از نوع adjency matrix کند .

```
def make_matrix(data,max):
    zero_array=np.zeros((max+1,max+1))
    counter=0
    for i in data:
        for j in i:
            zero_array[counter][j]=1
        counter+=1
    return zero_array
```

حالا ماتریس Adjency matrix را داریم و هر کاری خواستیم میتوانیم روی ان انجام دهیم .

میتوانیم با network گراف ان را رسم کنیم و یا میتوانیم با gephi ان را رسم کنیم در قسمت اخر عکس اول مقدار این ماتریس رو در یک فایل csv ذخیره کردم تا بتوانم از ان استفاده کنم .

```
import networkx as nx
G = nx.from_numpy_array(pagerank_array, create_using=nx.Graph)
pos = nx.layout.spring_layout(G)
nx.draw(G, node_size=1, width=0.0001)
```

در کد بالا ان را توسط network نوشتم که رسم کنم ولی ارور حافظه میدهد و توانایی رسم را ندارد بعد 1 ساعت رسم میکند ولی شبکه قابل تشخیص نیست.

```
sport = np.zeros(1001).reshape(1001, -1)
news = np.zeros(1001).reshape(1001, -1)
fun = np.zeros(1001).reshape(1001, -1)

counter=0
for i in data['websites']:
    if(i['category']=="news"):
        news[counter]=1
    if(i['category']=="sport"):
        sport[counter]=1
    if(i['category']=="fun"):
        fun[counter]=1
    counter+=1
```

در کد بالا اومدم بر اساس دسته بندی ها و شاخه category در داده ها هر شاخه ای که مربوط به کدام بخش است را پیدا کردم و در 3 وکتور ان را قرار دادم.

تا اگر کاربر برای مثال دسته بندی ورزشی را سرچ کرد به جای e/n معمولی وکتور مخصوص ان را قرار دهیم .

در بخش اول سوال : page rank مطرح شده است کد ان به صورت زیر است :

از کتابخانه scipy برای افزایش سرعت ضرب استفاده کردم این کتابخانه خانه های صفر را نگه نمیدارد .

```
import numpy as np
import pandas as pd
from scipy import sparse

adjacency_matrix_T = pagerank_array.T

# Creating M
M = np.zeros(adjacency_matrix_T.shape)
for c in range(adjacency_matrix_T.shape[1]):
    s = adjacency_matrix_T[:, c].sum()
    if(s!=0):
        M[:, c] = adjacency_matrix_T[:, c]/s
# Convertin M to csc matrix
M_csc = sparse.csc_matrix(M)
# Creating V matrix
v = np.zeros(1001).reshape(1001, -1)
# Convertin V to csc matrix
v_csc = sparse.csc_matrix(v)
# Creating e / n matrix
e = np.ones(1001).reshape(1001, -1)
n = 1001
e_n = e / n
e_n_csc = sparse.csc_matrix(e_n)
```

برای حل این الگوریتم چند تا متغیر اولیه لازم است چون فرمول ان به صورت زیر است .

$$B*(MV)+(1-B)e/n$$

با توجه به فرمول بالا در حالت عادی به جای e/n یک وکتور که همه مقادیر ان 1/1000 است را قرار میدهیم .

و به جای B به گفته گوگل بهترین مقدار 0.85 است که ان را قرار میدهیم .

ولی در قسمت دوم سوال به جای e/n مقدار وکتور اختصاصی ان را قرار میدهیم . که در قسمت قبل ان ها را پیدا کردیم .

کد بالا به ترتیب اول تولید وکتور ها است بعد با کتابخانه spacy , ماتریس اولیه خود را به ان داده تا فقط مقادیر غیر صفر را نگه دارد و عمل ضرب را روی ان ها انجام دهد .

در قسمت زیر عملیات اصلی این الگوریتم را انجام میدهیم .

```
d = 1
beta = 0.85
alpha = 0.0001
while(d>alpha):
    v_prime = beta * (M_csc * v_csc) + (1-beta) * e_n_csc
    #rescaling v_prime to sum of 1
    t = 1 / v_prime.sum()
    v_prime = v_prime * t
    v_d = abs(v_prime.toarray()-v_csc.toarray())
    d = v_d.sum()
    v_csc = v_prime

df_v = pd.Series(v_prime.toarray().ravel())
print("Top ten v=", df_v.nlargest(n=10))
```

```
Top ten v= 91     0.003253
201    0.003250
278    0.002729
202    0.002478
125    0.002399
71     0.002157
44     0.002147
313    0.002070
237    0.001932
86     0.001911
dtype: float64
```

عملیات فرمول بالا رو با کد در شکل بالا انجام دادم 10 تا بهترین رتبه رو نمایش دادم که شکل بالا نمایان گر ان است .

قسمت دوم سوال :

در این قسمت از کاربر یک متن گرفته و میگیرم که جزو کدام دسته است :

```
In [5]: word=input("get one of category : sport , news , fun  :")

        get one of category : sport , news , fun  :fun

In [6]: word

Out[6]: 'fun'
```

در زیر همان الگوریتم بالا رو بر اساس این قسمت e/n اختصاصی داده ام این به این معنی است که اگر کاربر اخبار سرچ کند بهش صفحه های خبری بیشتر پیشنهاد میدهد و ...

در زیر کد این قسمت اورده شده است .

```
# Convertin M to csc matrix
M_csc = sparse.csc_matrix(M)
# Creating V matrix
v = np.zeros(1001).reshape(1001, -1)
# Convertin V to csc matrix
v_csc = sparse.csc_matrix(v)
# Creating e / n matrix
e = np.ones(1001).reshape(1001, -1)
n = 1001
if(word=='fun'):
    n=np.count_nonzero(fun)
    e_n=fun/n
if(word=='sport'):
    n=np.count_nonzero(sport)
    e_n=sport/n
if(word=='news'):
    n=np.count_nonzero(news)
    e_n=news/n
e_n_csc = sparse.csc_matrix(e_n)

d = 1
beta = 0.85
alpha = 0.0001
while(d>alpha):
    v_prime = beta * (M_csc * v_csc) + (1-beta) * e_n_csc
    #rescaling v_prime to sum of 1
    t = 1 / v_prime.sum()
    v_prime = v_prime * t
    v_d = abs(v_prime.toarray()-v_csc.toarray())
    d = v_d.sum()
    v_csc = v_prime

df_v = pd.Series(v_prime.toarray().ravel())
print("Top ten v=", df_v.nlargest(n=10))
```

```
Top ten v= 201    0.003667
91     0.003313
278    0.002784
202    0.002453
71     0.002310
125    0.002221
237    0.002169
313    0.001915
82     0.001910
63     0.001888
dtype: float64
```

تفاوت کد بالا با قسمت اول فقط در بخش موضوعی بودن ان است و در اخر هم 10 صفحه پر بازدید را نمایش میدهد .

در قسمت سوم این سوال از الگوریتم HITS استفاده شده است .

کد ان به صورت زیر اورده شده است .

و در اخر بر اساس h,a 10 صفحه برتر نیز نمایش داده شده است .

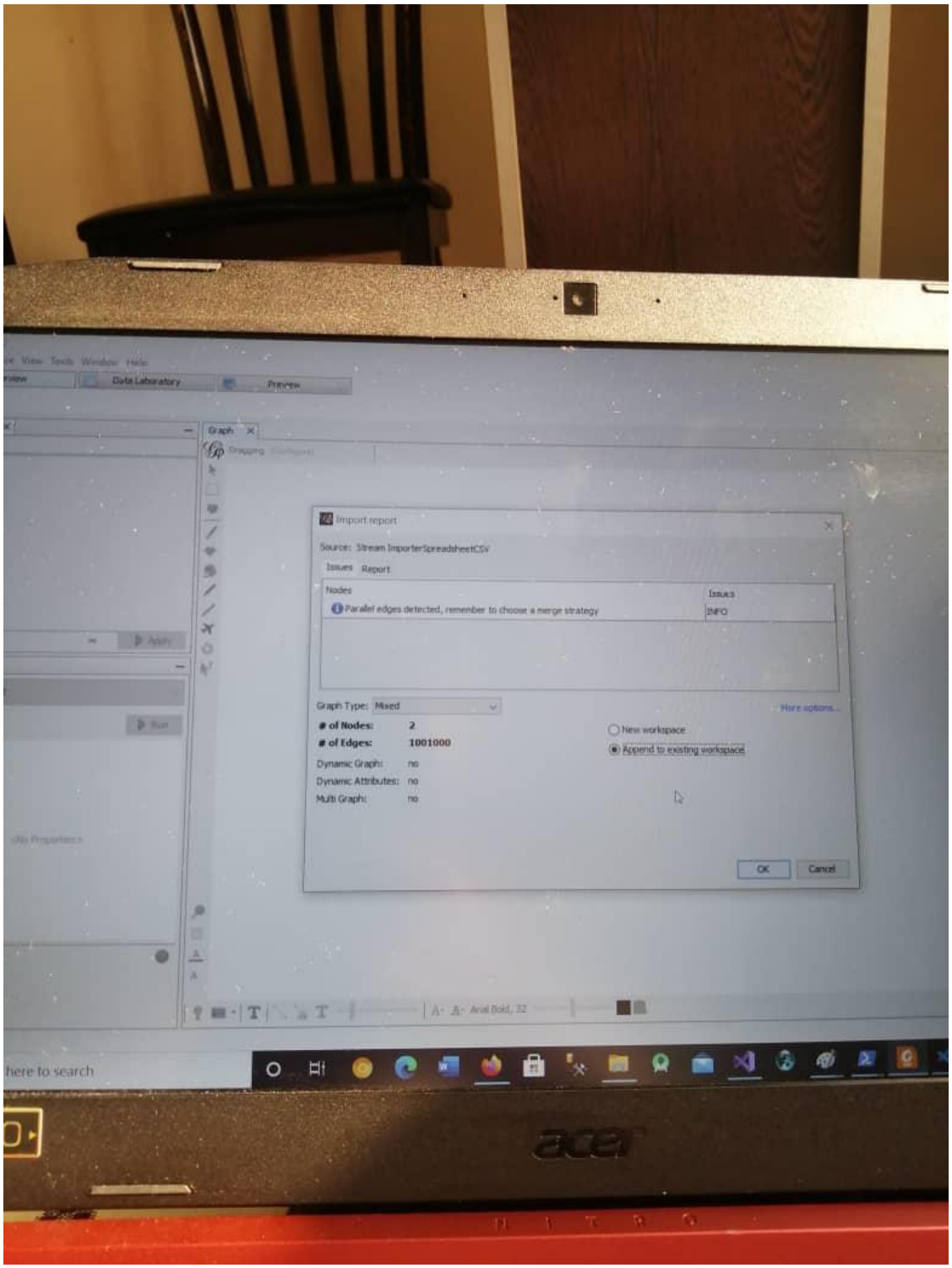
```
# HITS
# Creating link matrix
L = pagerank_array
# Converting link matrix to csc matrix
L_csc = sparse.csc_matrix(L)
# Creating h matrix
h = np.ones(1001).reshape(1001, -1)
# Converting h matrix to csc matrix
h_csc = sparse.csc_matrix(h)

d = 1
alpha = 0.001
while(d>alpha):
    a_csc = L_csc.transpose() * h_csc
    h_prime = L_csc * a_csc
    #rescaling h_csc to sum of 1
    t = 1001 / h_prime.sum()
    h_prime = h_prime * t
    h_d = abs(h_prime.toarray()-h_csc.toarray())
    d = h_d.sum()
    h_csc = h_prime

df_h = pd.Series(h_prime.toarray().ravel())
df_h.nlargest(n=10)
print("Top ten h=", df_h.nlargest(n=10))
df_a = pd.Series(a_csc.toarray().ravel())
df_a.nlargest(n=10)
print("Top ten a=", df_a.nlargest(n=10))
```

```
Top ten h= 929    1.964874
413    1.952038
459    1.943605
343    1.942080
204    1.938160
518    1.935616
828    1.930222
796    1.927166
546    1.925466
467    1.920444
dtype: float64
Top ten a= 63     281.107309
237    268.819060
71     267.460137
202    263.811741
125    251.529360
278    247.960997
91     247.790908
44     244.829217
187    244.637684
201    243.839791
dtype: float64
```

در قسمت چهارم : فایل تولید شده csvرو دادم به گفی memory error میده اجرا نشد .



تمرین سوم :

۳-ماتریس وِب M را با ابعاد $n \times n$ (n تعداد صفحات وب است) در نظر بگیرید. هر ورودی m_{ij} در سطر i و ستون j برابر با صفر است، مگر اینکه یالی از گره (صفحه) j به گره i وجود داشته باشد. در این صورت مقدار m_{ij} برابر با $1/k$ است و k تعداد یالهای خروجی گره j است. پس اگر گره j دارای $k > 0$ یال خروجی باشد، ستون j دارای k مقدار $1/k$ خواهد بود و بقیه درایه‌ها نیز صفر هستند. اگر گره j یک گره بَن‌بست باشد (یال خروجی نداشته باشد) همه درایه‌های ستون j صفر خواهد بود.

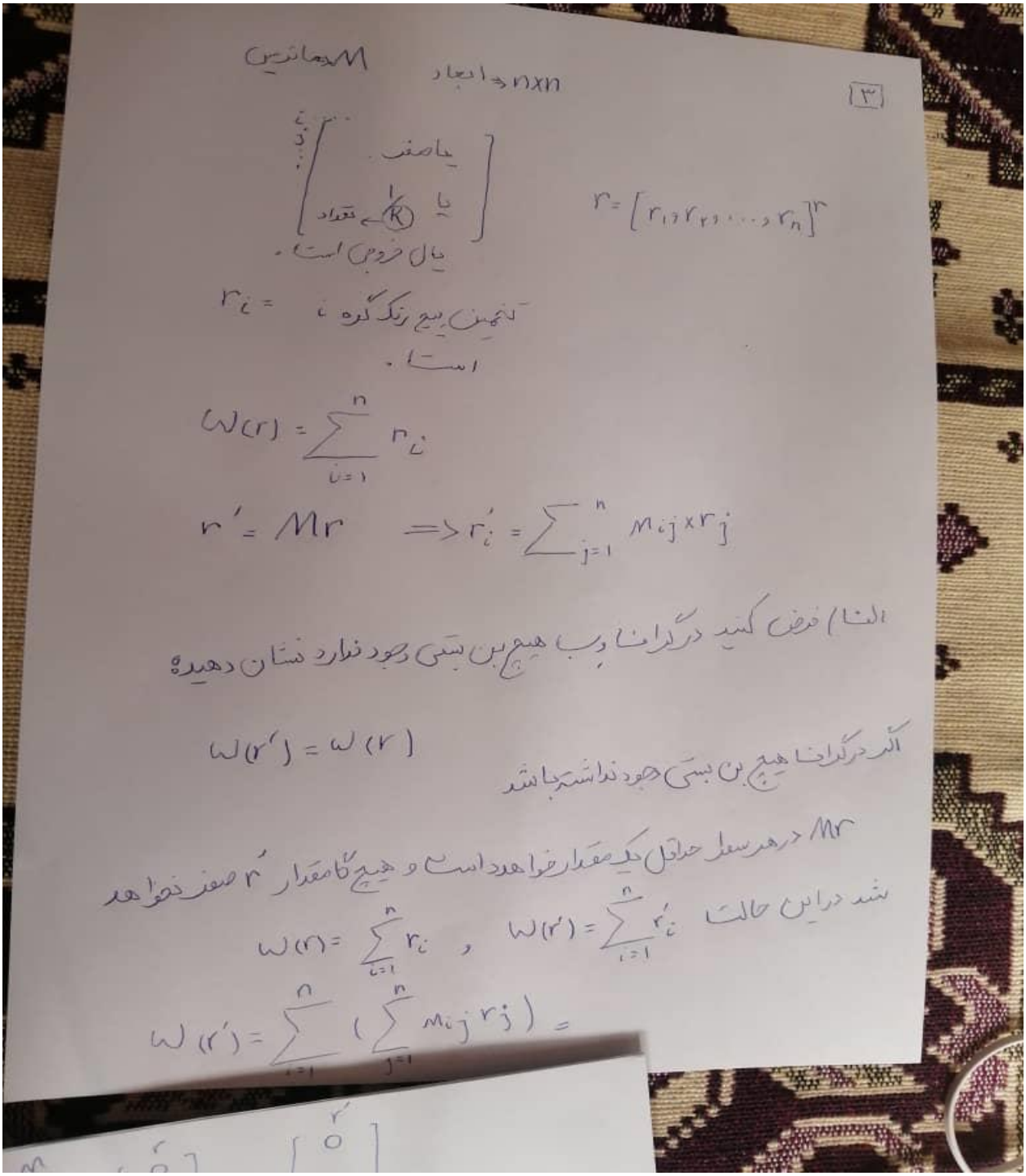
فرض کنید $r = [r_1, r_2, \dots, r_n]^T$ بردار پیج‌رَنک باشد. به این معنی که r_i تخمینی از پیج‌رَنک گره i است. فرض کنید $w(r)$ حاصل جمع مولفه‌های بردار r باشد، یعنی $w(r) = \sum_{i=1}^{n} r_i$.

در هر تکرار از الگوریتم، تخمین بعدی از بردار پیج‌رَنک یعنی r' به صورت $r' = Mr$ محاسبه می‌شود. به عبارت دیگر برای هر i خواهیم داشت $r'_i = \sum_{j=1}^{n} M_{ij} r_j$. همچنین فرض می‌کنیم که $w(r')$ برابر با مجموع مولفه-های r' باشد، یعنی $w(r') = \sum_{i=1}^{n} r'_i$.

الف) فرض کنید در گراف وِب هیچ بَن‌بستی وجود ندارد. نشان دهید $w(r') = w(r)$.

ب) فرض کنید در گراف وِب هیچ بَن‌بستی وجود ندارد اما ما از یک احتمال teleportation برابر با $1 - \beta$ استفاده می‌کنیم به این معنی که با این احتمال به یک گره تصادفی خواهیم رفت $(0 < \beta < 1)$. بنابراین تخمین بعدی از r_i برابر خواهد بود با $r'_i = \beta(\sum_{j=1}^{n} M_{ij} r_j) + (1 - \beta)/n$. تحت چه شرایطی $w(r') = w(r)$ خواهد بود؟ ثابت کنید[۲].

من فرق اینو با page rank معمولی خودمون که taxation هم داره نتونستم از صورت سوال بفهمم.


M همانتین
ابعاد ⇒ n×n
۳
یا صفر
یا $\frac{1}{K}$ ← تعداد
یال خروجی است.
$r = [r_1, r_2, \dots, r_n]^T$
$r_i =$ تخمین پیج رنک گره i است.
$W(r) = \sum_{i=1}^{n} r_i$
$r' = Mr \Rightarrow r'_i = \sum_{j=1}^{n} M_{ij} \times r_j$
الف) فرض کنید در گراف وب هیچ بن بستی وجود ندارد نشان دهید:
$W(r') = W(r)$
اگر در گراف هیچ بن بستی وجود نداشته باشد
Mr در هر سطر حداقل یک مقدار خواهد است و هیچ گاه مقدار r صفر نخواهد
شد در این حالت $W(r) = \sum_{i=1}^{n} r_i$ ، $W(r') = \sum_{i=1}^{n} r'_i$
$W(r') = \sum_{i=1}^{n} \left(\sum_{j=1}^{n} M_{ij} r_j \right) =$