به نام خداوند بخشنده و مهربان

توضیح پروژه درس ساختمان داده

استاد درس : جناب آقای دکتر دهقان تخت فولادی

تدریس یاران : جناب آقای مهندس احمدپناه و جناب آقای مهندس اسدی

موضوع پروژه : تشخیص ساختارهای اجتماعی در گراف

دانشجو : روزبه قاسمی ۹۵۳۱۴۲۴

پاییز ۱۳۹۶

توضیح پروژه دوم

مقدمه

در این پروژه قصد داشتیم نشان دهیم که ساختار های اجتماعی گراف هارا چگونه می‌توانیم نشان دهیم .

این پروژه نحوه‌ی کارکرد شبکه های اجتماعی (Social Networks) را بررسی می‌کنیم . یعنی به طور دقیق‌تر در این پروژه هر گره را به عنوان یک کاربر یا User در نظر می‌گیریم که برای ارتباط با بقیه کاربران از طریق یال که تنها پل ارتباطی هر کاربر با بقیه کاربران است ، استفاده می‌کنیم . در این پروژه از User به جای Vertex یا راس استفاده می‌کنیم .

برای شروع ابتدا فایل مورد نظر را می‌خوانیم سپس یک بار با لیست همسایگی و یک بار با ماتریس همسایگی آن را ذخیره می‌کنیم . سپس برای هر کدام از این روش های ذخیره ساختمان های داده ، الگوریتم های پیمایش را پیاده می‌کنیم.

ابتدا الگوریتم Quick Sort پس از آن الگوریتم Insertion Sort و بعد الگوریتم Bubble Sort و در نهایت الگوریتم Optimum Sort که در نهایت به این نتیجه می‌رسیم که این الگوریتم ، بهینه ترین الگوریتم نسبت به بقیه الگوریتم هاست ، را برای هر کدام پیاده می‌کنیم.

در نهایت با استفاده از مزیَت زبان برنامه‌نویسی Matlab که قادر به نمایش نمودار است ، استفاده کرده و نمودار بدست آمده را تحلیل می‌کنیم.

مقایسه مشخصات سخت افزاری

	Mine	Friend
CPU Model	**Intel Core i7 6700HQ**	
CPU Physical Core	**7.89 GB**	
CPU Virtual Core	**2.92 GB**	
CPU L1 Cash	**128**	
CPU L2 Cash	**1024**	
CPU L3 Cash	**6144**	
RAM Model	**DDR4**	
RAM Capacity (GB)	**64**	
RAM Bus	**2133 MHZ**	
HDD/SSD Write Speed	**114.33 MB**	
HDD/SSD Read Speed	**123.98 MB**	
OS	**Microsoft Windows10 64 Bit Pro**	

توضیحات تکمیلی جدول :

از آنجایی که اکثراً نرم افزار متلب در سیستم های خود نداشته اند ، برای همین با کامپیوتر های سایت دانشکده یکبار دیگر تست هارا انجام دادم.

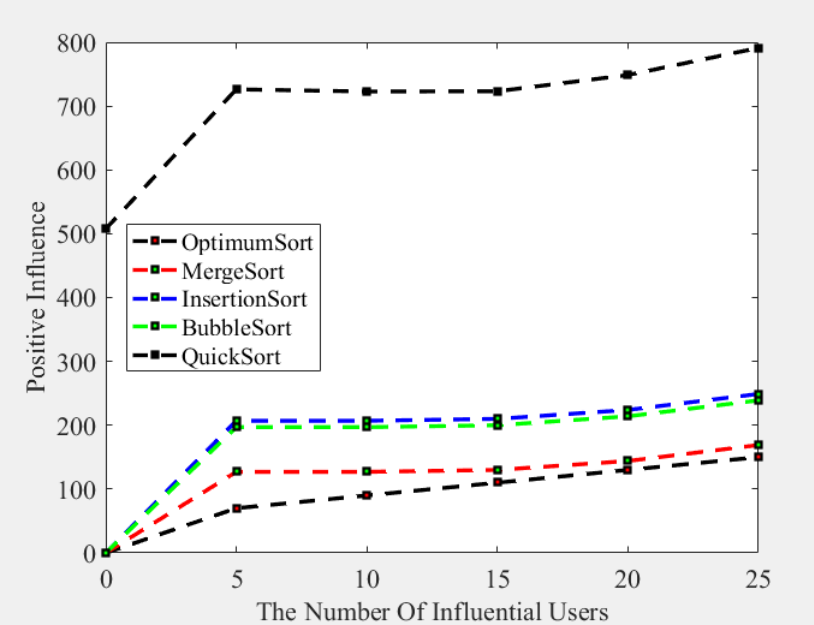
جدول نتایج

Name	Vertex Number	Edge Number	Average Vertex Degree
Test1	۱۰۰۰۰	۷۷۰۴۸	۱۵.۴۰۹۶
Test2	۱۰۰۰۰	۱۲۶۹۰۳	۲۵.۳۰۸۶
Test3	۵۰۰۰۰	۳۸۲۵۵۹	۱۵.۳۰۲۳۶
Test4	۵۰۰۰۰	۶۳۴۵۵۳	۲۵.۳۸۱۲
Test5			
Test6			
Test7			
Test8			

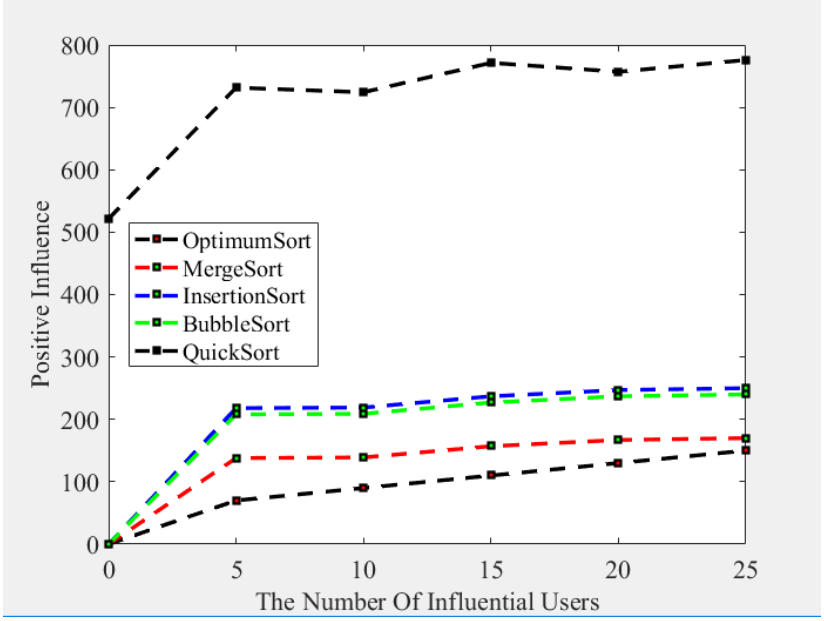
مشاهدات و نتایج :

با افزایش تعداد رئوس تعداد یال ها هم به طبع افزایش می‌یابد اما متوسط درجه رئوس بستگی به اندازه یال‌ها دارد.

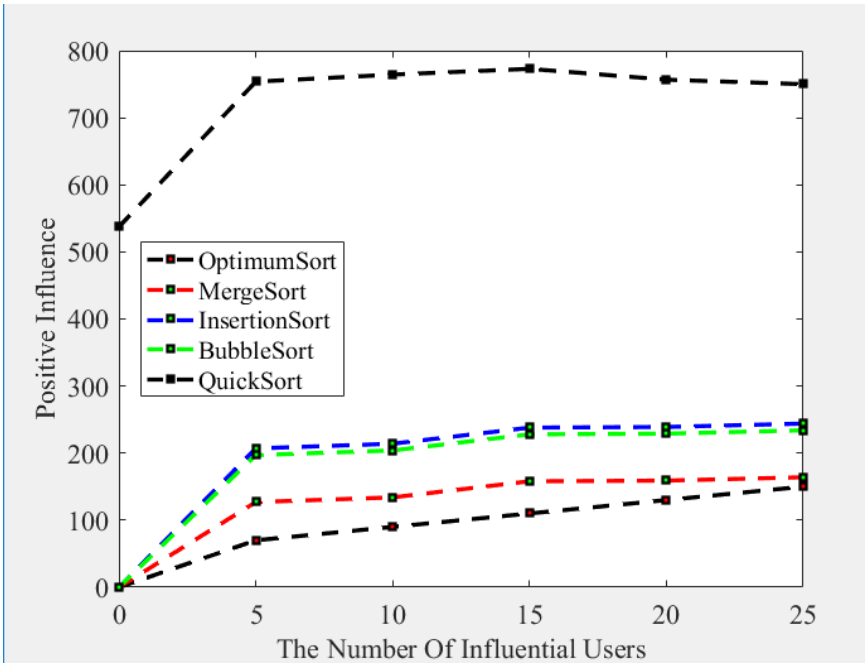
نمودارها



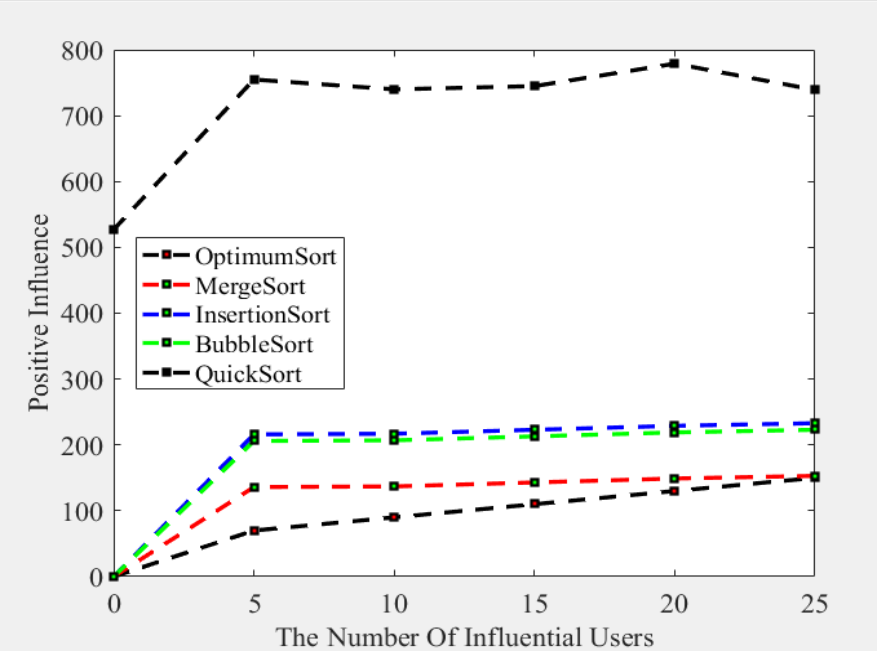
نمودار خروجی برای تست کیس شماره ۱



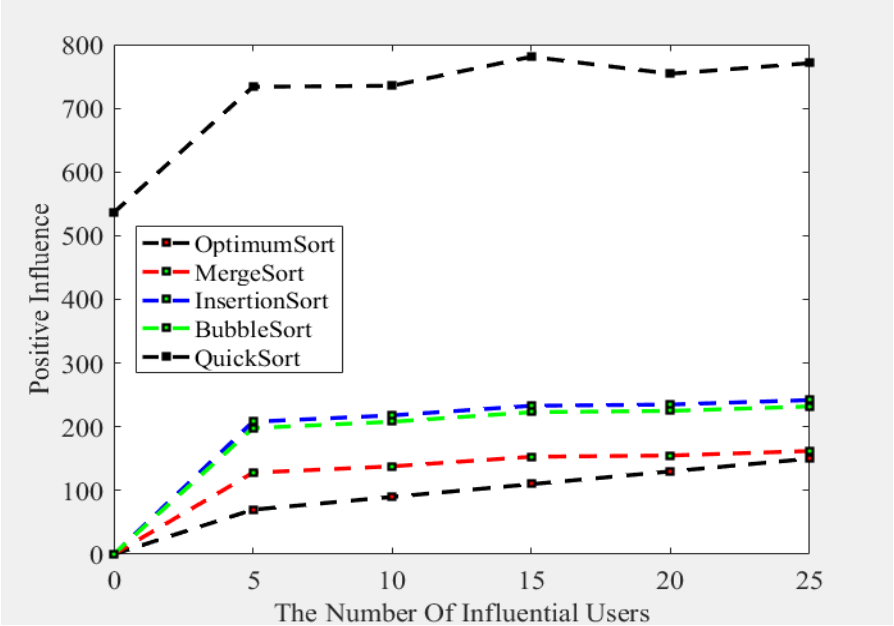
نمودار خروجی برای تست کیس شماره ۲



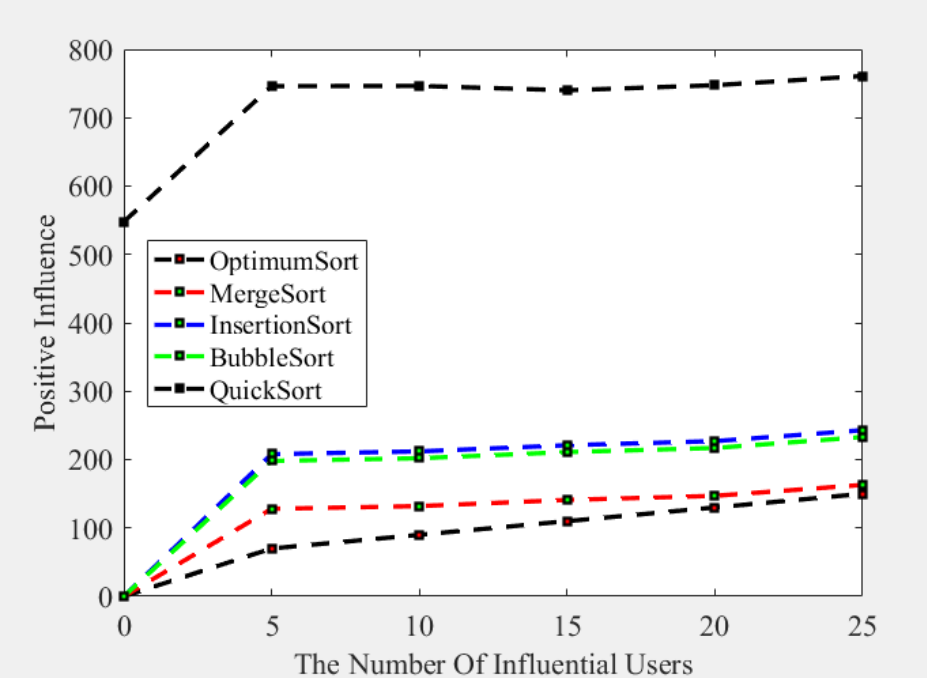
نمودار خروجی برای تست کیس شماره ۳



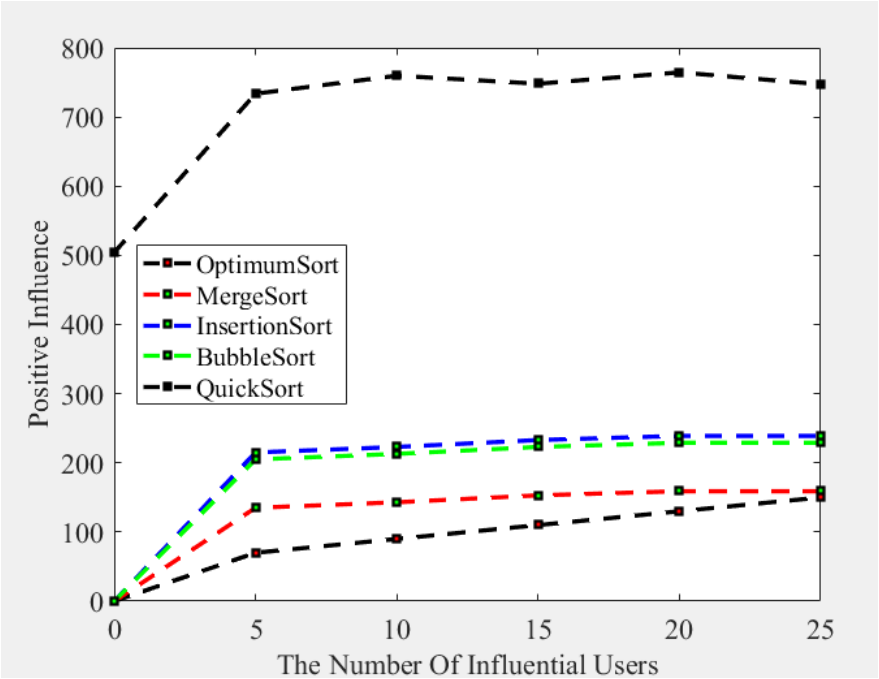
نمودار خروجی برای تست کیس شماره ۴



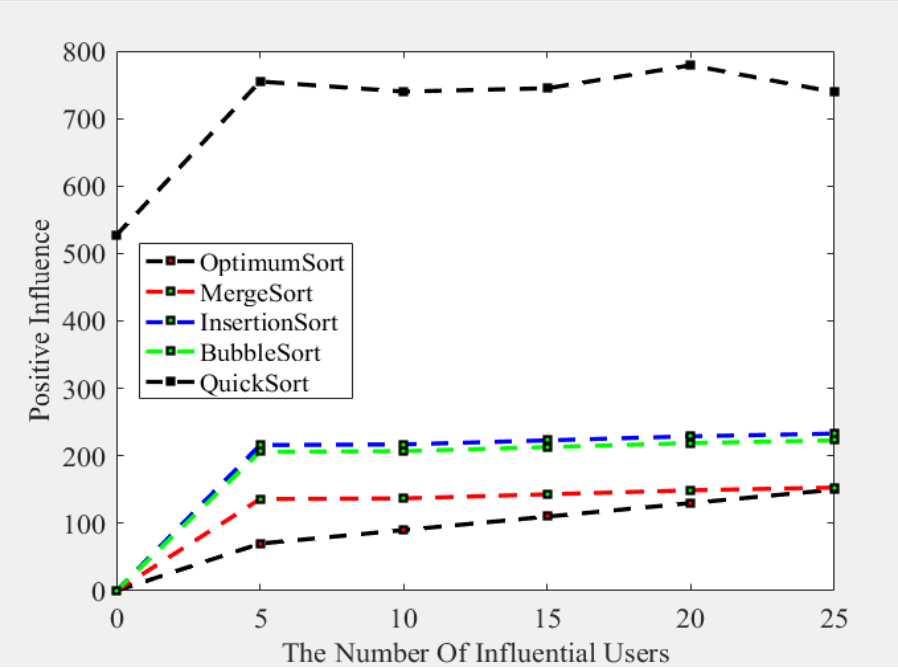
نمودار خروجی برای تست کیس شماره ۵



نمودار خروجی برای تست کیس شماره ۶



نمودار خروجی برای تست کیس شماره ۷



نمودار خروجی برای تست کیس شماره ۸

نحوه‌ی اجرای برنامه

برای اجرای برنامه نام فایل مورد نظر را که در همان Folder ای هست که سورس برنامه در آن هست را به برنامه می‌دهیم سپس برنامه شروع به اجرا می‌شود و تمام الگوریتم ها را روی فایل خوانده شده پیاده می‌کنیم.

سپس برای هرکدام روی یک نمودار پیاده می‌شود و در نتیجه به این موضوع میرسیم که Optimum Sort بهینه ترین الگوریتم است.

از آنجایی که اجرای تمام دستورات باعث طولانی شدن توضیح پروژه می‌شود ، فقط چند دستور ابتدایی داده و خروجی آن هارا نشان می‌دهیم :

```
%vorude data
Data=importdata('test1.txt');
```

از آنجایی که برای هر سورت برنامه را اجرا میکنیم برای همین چندین بار فرآیند حذف یال با کوچکترین مقدار را انجام می‌دهد . در نتیجه به ازای هر بار پیمایش و حذف یک سری یال ، خروجی های مناسب خود را نمایش می‌دهد.

```
The Number of Influential Users: 5
S = 9320
Users QSS(MyApproach)* = 2435  4858  5571  5795  6343  6349  6867  7320  7729  7979  8067  8371  8399  8510  8592  8599  8624  8664  8746  8791  8
Users QSS(Random)* = 2435  4858  5571  5795  6343  6349  6867  7320  7729  7979  8067  8371  8399  8510  8599  8664  8746  8791  8881  9209  9214
    35    35    60    80   100
    25   100   125    55    20
   150   120   100    60   110
    50    70    40   140    40
    90   110    90    50    30
     0   110    90    40    50
    60    70     0   110    10
   140    70    90   100    30
    10   130   130    40   140
    30    90    70    40     0
    70   100   100    50    90
   110    50    50   110   100

    30    35    25    40    55

         100          400          800         1100         1700
        1500         1800         1200         2200         1500
        1400         1000          400         1000          700
         800          700          900          700         1200

        4500         2000         2500         3000

   300   500   250   150
```

جمع‌بندی

پس از اجرای پروژه بر روی سیستم های مختلف به این نتیجه می‌رسیم که هرچه میزان Ram و CPU Cache بیشتر باشد سرعت پردازش بیشتر می‌شود و مقدار N متفاوت می‌شود . همچنین از نمودار های خروجی می‌توان به این نتیجه رسید که سرعت تغییرات در الگوریتم های Quick Sort و Merge Sort تقریبا باهم برابر بوده و همچنین الگوریتم های Bubble Sort و Insertion Sort سرعت رشد تغییرات یکسانی دارند .

از طرفی الگوریتم Optimum Sort به وضوح اختلاف قابل توجهی با بقیه دارد و توانایی بیشتری در پردازش دارد .یکی از مهم ترین مسائل سرعت پردازش سیستم هاست. به وضوح سرعت پردازش در سیستم شخصی خودم بالاتر از کامپیوتر های سایت دانشکده بود!

برای محاسبه N می‌گوییم آن n هایی را بدست آور که از بهترین حالت الگوریتم Quick Sort و حالت متوسط آن بهتر است. در نتیجه با قرار دادن n < nlogn برای بهترین حالت و n^2 > nlogn برای حالت متوسط مقدار N مطلوب را بدست آوریم.

الگوریتم	بهترین حالت	حالت متوسط	بدترین حالت
Bubble Sort	n	n^2	n^2
Insertion Sort	n	n^2	n^2
Quick Sort	nlogn	nlogn	n^2