به نام خدا

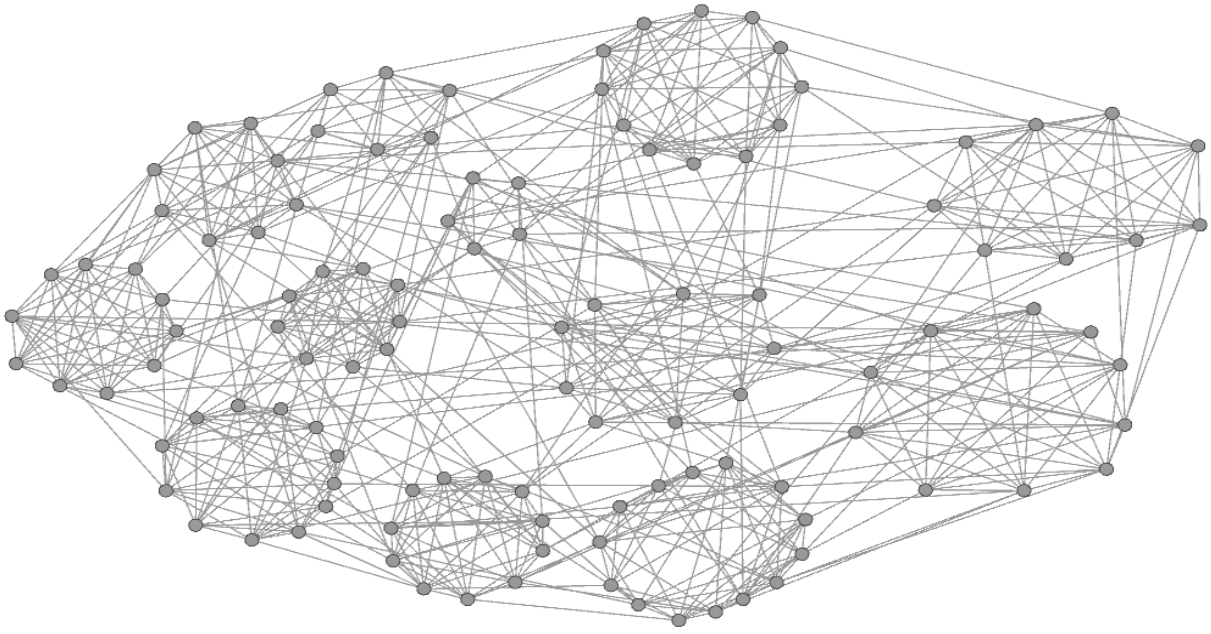
پروژه دوم درس ساختمان داده‌ها

تشخیص ساختارهای اجتماعی در گراف

مقدمه

هدف این پروژه، پیاده‌سازی یک الگوریتم مربوط به گراف‌ها است. این الگوریتم در سال ۲۰۰۴ ارائه شده است که ساختارهای اجتماعی را در یک گراف تشخیص می‌دهد. مقاله و ترجمه آن به زبان فارسی در پیوست تعریف پروژه قرار دارد. با انجام آزمایش‌هایی روی پیاده‌سازی انجام‌شده از این الگوریتم، باید میزان حافظه مصرفی و زمان اجرا را برای گراف‌های با اندازه‌های مختلف محاسبه نمایید. همچنین، شما باید تاثیر الگوریتم‌های متفاوت مرتب‌سازی را در زمان و حافظه مصرفی محاسبه کنید. در ادامه، الگوریتم مورد نظر و جزئیات تعریف پروژه آمده است.

اجتماع در یک گراف به زیرگرافی گفته می‌شود که تعداد یال‌های بین رأس‌های داخلی زیرگراف نسبت به یال‌های خارج‌شده از آن زیرگراف (به رئوس دیگر موجود در گراف) بیشتر باشد. به عنوان مثال، شکل زیر اجتماع‌های موجود در گراف بازی‌های فوتبال را نشان می‌دهد.



شرح الگوریتم

در این الگوریتم، با حذف یال‌های بین اجتماع‌ها، گراف به زیر گراف‌های متراکم‌تر تقسیم می‌شود. معیاری که این الگوریتم برای امتیازدهی به یال‌ها استفاده می‌کند، عبارت است از:

$$C_{ij} = \frac{z_{ij} + 1}{\min[k_i - 1, k_j - 1]}$$

که در آن z_{ij} برابر با تعداد دورهای ساده به طول سه است که یال بین رأس i و j در آن دور دیده می‌شود. بدیهی است که اگر یالی بین دو رأس i و j وجود نداشته باشد، مقدار C_{ij} برای آن‌ها تعریف نمی‌شود. همچنین، k_i درجه رأس i و k_j درجه رأس j است. بنابراین، اگر درجه یکی از رأس‌ها عدد یک باشد، آنگاه مخرج کسر برابر با صفر می‌شود که تعریف نشده است؛ یعنی در این حالت، مقدار C_{ij} را باید برابر با عددی بسیار بزرگ در نظر بگیرید.

قدم‌های الگوریتم به شرح زیر است:

۱- محاسبه امتیاز C_{ij} برای تمام یال‌ها
۲- مرتب‌سازی صعودی یال‌ها بر اساس امتیاز محاسبه‌شده
۳- حذف یال با کوچکترین مقدار C_{ij} از گراف
۴- اگر گراف به دو بخش تقسیم شده است، آنگاه پایان الگوریتم
۵- برو به قدم اول

ورودی برنامه فایلی در قالب CSV[1] است که در آن، هر رأس با یک عدد طبیعی نشان داده می‌شود و بیانگر گراف ورودی است. تمام فایل باید یک‌جا خوانده شده و در حافظه RAM بارگذاری شود. جهت ذخیره گراف در حافظه، باید از دو ساختمان داده ماتریس همسایگی و لیست همسایگی استفاده کنید. جهت مرتب‌سازی نیز باید از الگوریتم‌های Quick Sort، Insertion sort، Merge Sort، Bubble sort و Optimum sort (که در ادامه شرح داده خواهد شد) استفاده شود. قبل از اجرای الگوریتم، باید مشخص شود که گراف با چه ساختمان داده‌ای ذخیره شود و چه الگوریتمی برای مرتب‌سازی انتخاب شده است. برنامه پس از گرفتن آدرس فایل گراف ورودی، باید دستورات زیر را از کاربر دریافت کند:

1) RUN LinkedList Quick

2) RUN Matrix Quick

[1] Comma-Separated Values

3) RUN LinkedList Insertion

4) RUN Matrix Insertion

5) RUN LinkedList Merge

6) RUN Matrix Merge

7) RUN LinkedList Bubble

8) RUN Matrix Bubble

9) RUN LinkedList Optimum Insertion N

10) RUN Matrix Optimum Insertion N

11) RUN LinkedList Optimum Bubble N

12) RUN Matrix Optimum Bubble N

قالب کلی دستورات مطرح‌شده این‌گونه است که بعد از کلمه RUN، نوع ساختمان داده ذخیره‌سازی تعیین و سپس، نوع الگوریتم مرتب‌سازی مشخص می‌شود. منظور از الگوریتم مرتب‌سازی Optimum این است که مرتب‌سازی بر اساس الگوریتم Quick انجام می‌شود؛ با این تفاوت که اگر اندازه یک زیرآرایه در طول اجرا، کمتر از N (عددی که کاربر به جای N وارد کرده است) باشد، آن زیرآرایه با روش Bubble یا Insertion مرتب‌سازی شود. هدف از این نوع مرتب‌سازی، بهبود کارایی و سرعت مرتب‌سازی است. زیرا زمانی که طول آرایه خیلی کوچک باشد، فراخوانی بازگشتی Quick باعث افت سرعت می‌شود. معمولا مقدار N کمتر از ۳۰ است، ولی این عدد به مشخصات سخت‌افزاری سیستم مانند حافظه نهان (Cache) وابسته است. یکی از موارد این پروژه، یافتن این عدد برای سیستم شما است.

همان‌گونه که گفته شد، گراف‌های ورودی به صورت CSV (با فرمت فایل txt) هستند که هر سطر آن، یک جفت رأس (بیانگر یک یال بدون جهت) است. پس از تعیین فایل ورودی، کاربر می‌تواند دستورات بالا را وارد کند. خروجی این برنامه باید حاوی اطلاعات زیر باشد:

۱- زمان مصرف‌شده برای خواندن گراف از دیسک و ذخیره آن در ساختمان داده تعیین‌شده
۲- زمان اجرای الگوریتم (پنج قدم ذکرشده)
۳- حافظه مصرفی توسط الگوریتم، با توجه به ساختمان داده‌های استفاده‌شده (می‌توانید از ابزارهای نظارت بر مقدار حافظه مصرفی مانند JMX استفاده کنید و از نمودار آن اسکرین‌شات بگیرید.)
۴- یک فایل به نام result.txt که حاوی چنین اطلاعاتی است:

#VertexNum : A or B

یعنی در هر سطر از فایل نتیجه، شماره اندیس رأس و اجتماعی (دسته‌ای) که در آن قرار دارد مشخص می‌شود. (به عنوان مثال، B : 2#) فرض بر این است که رأس اندیس شماره ۱ همواره در اجتماع A قرار دارد (یعنی A : 1#).

۵- زمان اجرای کل برنامه

قالب گزارش

گزارش باید تایپ شده، در قالب PDF و به زبان فارسی باشد. در ابتدای گزارش، در یک پاراگراف (چکیده) کلیت گزارش خود را شرح دهید. در دو پاراگراف (مقدمه) تعریف و کاربرد تشخیص اجتماع‌ها را توضیح دهید. در حداقل سه پاراگراف (شرح الگوریتم)، نحوه پیاده‌سازی، ساختمان داده‌های استفاده‌شده، الگوریتم‌های استفاده‌شده (برای تشخیص دور به طول سه و تشخیص تقسیم گراف به دو بخش مجزا) و خود الگوریتم را کاملا شرح دهید. همچنین، باید فلوچارت الگوریتم و پیچیدگی زمانی و حافظه‌ای آن را نیز تحلیل کنید. در یک پاراگراف (مشخصات سخت‌افزاری) باید مشخصات سیستم خود و یکی از دوستان خود را توضیح دهید. برای این کار ابزارهای مختلفی مانند CPU-Z وجود دارد. این بخش باید شامل جدول زیر باشد:

	Mine	Friend
CPU Model		
CPU Physical Core		
CPU Virtual Core		
CPU L1 Cash		
CPU L2 Cash		
CPU L3 Cash		
RAM Model		
RAM Capacity (GB)		
RAM Bus		
HDD/SSD Write Speed		
HDD/SSD Read Speed		
OS		

توجه شود که سیستم‌ها باید متفاوت باشند.

در یک پاراگراف (داده‌ها) باید داده‌های مسئله را معرفی نمایید. تعدادی فایل ورودی در اینجا قرار دارد. می‌توانید گراف‌های دیگری را نیز به عنوان داده ورودی اضافه کنید. این بخش باید شامل جدول زیر باشد:

Name	Vertex Number	Edge Number	Average Vertex Degree
test1	1000	3500	3.5

در حداقل ۵ پاراگراف (مقایسه و نتایج) مشاهده‌ها و نتایج خود را شرح دهید که شامل موارد زیر می‌شوند:

۱) تفاوت استفاده از ماتریس همسایگی و لیست همسایگی در زمان و حافظه (به صورت نمودارهای جداگانه برای هر سیستم و یک پاراگراف تحلیل برای هر نمودار)

۲) یافتن مقدار N ایده‌آل برای هر دو سیستم (به صورت نمودارهای جداگانه برای هر سیستم و یک پاراگراف تحلیل برای هر نمودار). توجه شود که این آزمایش را روی بزرگترین داده انجام دهید.

۳) مقایسه کارایی زمانی و حافظه‌ای الگوریتم روی دو سیستم

این بخش از گزارش باید حداقل حاوی نمودارهای زیر باشد:

- نموداری برای یافتن مقدار N ایده‌آل برای دستورهای شماره ۹ تا ۱۲؛ bubble

- نموداری برای مشخص‌کردن زمان صرف‌شده برای دستور شماره ۹ (با مقدار N ایده‌آل) در دو حالت لیست همسایگی و ماتریس همسایگی؛

- نموداری برای مشخص‌کردن حافظه مصرف‌شده برای دستور شماره ۹ (با مقدار N ایده‌آل) در دو حالت لیست همسایگی و ماتریس همسایگی؛ just t1 and t2

- نموداری برای مشخص‌کردن تأثیر تفاوت سخت‌افزار دو سیستم برای دستور شماره ۹ (با مقدار N ایده‌آل) در حالت لیست همسایگی روی تمام داده‌ها برحسب زمان کل اجرا (یک خط نمودار مربوط به سیستم اول و خط دیگر مربوط به سیستم دوم)؛

- نموداری برای مقایسه زمان کل اجرای برنامه با همه دستورات با شماره فرد (دستورات مبتنی بر لیست همسایگی) روی تمام داده‌ها

- نموداری برای مقایسه زمان اجرای پنج قدم الگوریتم با همه دستورات با شماره فرد (دستورات مبتنی بر لیست همسایگی) روی تمام داده‌ها؛

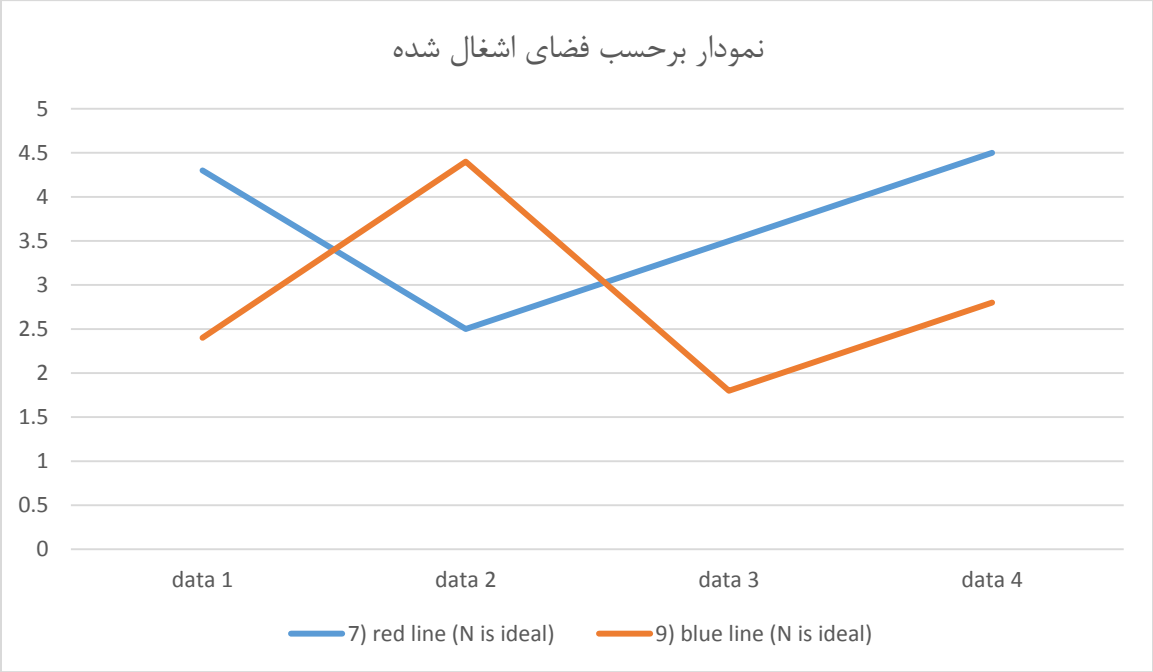
- نموداری برای مقایسه زمان کل اجرای برنامه با همه دستورات با شماره زوج (دستورات مبتنی بر ماتریس همسایگی) روی تمام داده‌ها؛

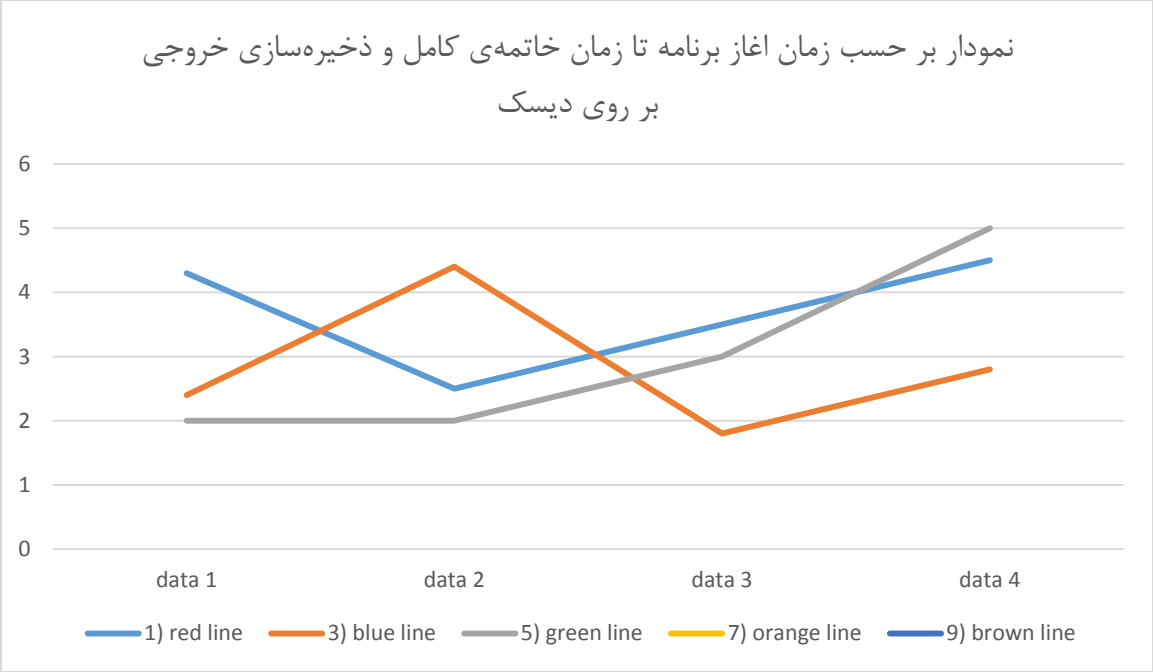
- نموداری برای مقایسه زمان اجرای پنج قدم الگوریتم با همه دستورات با شماره زوج (دستورات مبتنی بر ماتریس همسایگی) روی تمام داده‌ها؛

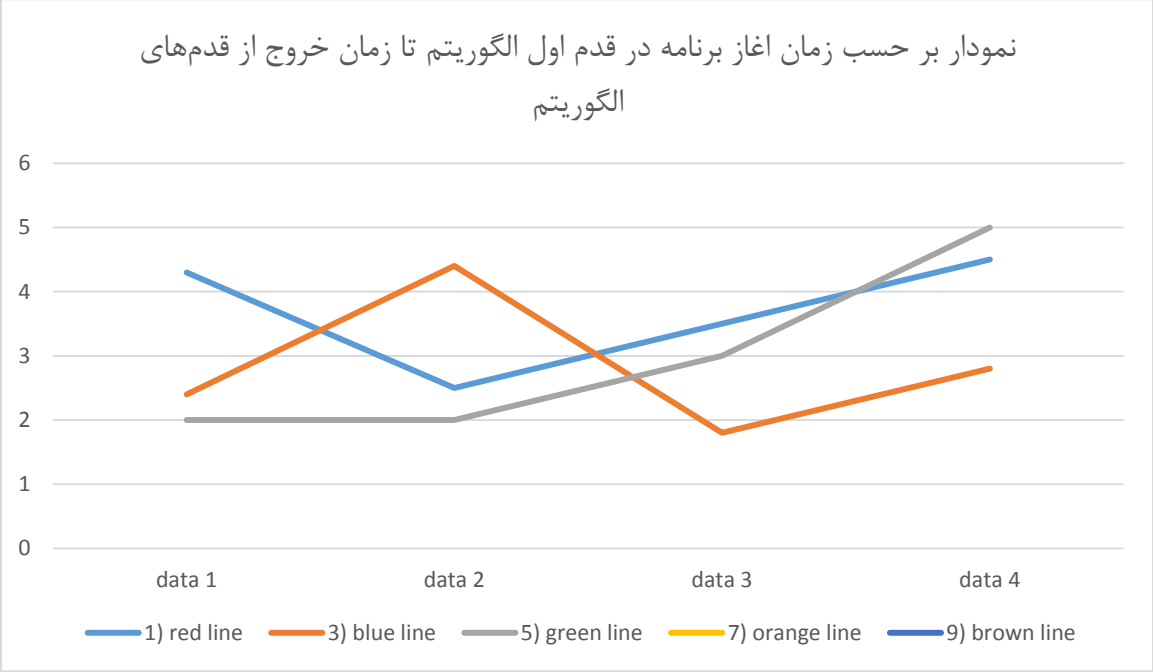
همچنین تمام این نمودارها برای سیستم دوم نیز رسم شود.

در دو پاراگراف (جمع‌بندی) تأثیر تفاوت سخت‌افزارهای مختلف و الگوریتم‌های مختلف را شرح داده و خلاصه کار خود را شرح دهید.

در ادامه نمونه‌هایی از نمودارهای مورد نظر آمده است.




نمودار بر حسب زمان اغاز برنامه تا زمان خاتمه‌ی کامل و ذخیره‌سازی خروجی بر روی دیسک
6
5
4
3
2
1
0
data 1
data 2
data 3
data 4
1) red line
3) blue line
5) green line
7) orange line
9) brown line


نمودار بر حسب زمان اغاز برنامه در قدم اول الگوریتم تا زمان خروج از قدم‌های الگوریتم
6
5
4
3
2
1
0
data 1
data 2
data 3
data 4
1) red line
3) blue line
5) green line
7) orange line
9) brown line

نکات پیاده‌سازی

در پیاده‌سازی این پروژه به موارد زیر پیاده‌سازی شود:

۱. همه ساختمان داده‌های مورد نیاز با توجه به تعریف پروژه باید پیاده‌سازی شوند. استفاده از ساختمان داده‌های آماده مجاز نیست. فقط استفاده از آرایه (ساده، ArrayList یا Vector) مجاز است. صف، پشته، لیست پیوندی، درخت یا هر ساختمان داده دیگر مورد استفاده باید پیاده‌سازی شود. (به عنوان درس توجه کنید!)

۲. پیاده‌سازی باید به صورت تک نفره باشد و محدودیتی برای زبان پیاده‌سازی وجود ندارد. اما دقت کنید که استفاده از ساختمان داده‌های آماده و به صورت کتابخانه مجاز نیست.

۳. در حین انجام پروژه، بحث و بررسی بین دانشجویان آزاد است اما هر دانشجو موظف است به تنهایی پروژه را انجام دهد و در هنگام تحویل حضوری، دانشجو باید به تمام جزئیات پیاده‌سازی کد کاملاً مسلط باشد. در مورد قسمت‌هایی از کد و نحوه عملکرد برنامه نیز از دانشجو سوال خواهد شد. همچنین با مواردی که تقلب و کپی‌کردن تشخیص داده شوند، برخورد جدی خواهد شد (برای تشخیص درصد شباهت کدها از سامانه Moss استفاده می‌شود).

۶. برای پرسش و پاسخ درباره پروژه فقط از طریق فروم موجود در سیستم مدیریت دروس استفاده کنید.

۷. موعد تحویل این پروژه تا ساعت ۲۳:۵۵ روز چهارشنبه ۱۳ دی ۱۳۹۶ خواهد بود. پوشه مربوط به کد پروژه را همراه با فایل pdf حاوی شرح انجام پروژه، نحوه اجرای برنامه و گزارش مربوط به تحلیل ساختمان داده‌های مورد استفاده و نیز فایل‌های تست را در قالب یک فایل zip به شکل زیر بارگذاری کنید. زمان و چگونگی نحوه تحویل حضوری متعاقباً اعلام می‌شود.

StudentNumber-FirstName-LastName-Project2.zip

e.g. 9531555-Ali-Ahmadi-Project2.zip