گزارش پروژه نهایی الگوریتم
نام:مهدی آفریده
شماره دانشجویی :9731131

:مقدمه

در این پروژه،ما می خواهیم الگوریتمی برای پیدان کردن community ها در گراف هایی همبند و غیر جهت را پیاده سازی کنیم.این الگوریتم با توجه به این به دست آمده است که در community ها،گره ها با هم رابطه زیادی دارند و هر گره با تعداد نسبتا قابل توجهی از گره های دیگر در ان community در ارتباط هست و برای همین،دور های زیادی بین گره های موجود در community وجود دارد،و همانطور که می دانیم،هر دور حداقل باید باید بین سه گره باشد،و در واقع ما می خواهیم با توجه به این نکته،community ها ،که در ان ها بیشترین تعداد overlap ها از دور های سه گره ای است را پیدا کنیم،و این کار را به صورت iterative با حذف یال هایی که وزن کمتری دارند،که این وزن از اینکه چه قدر در دور های سه گره ای حضور داشتند و اینکه آیا بین دو community می باشند(که با min گرفتن دو درجه گره های یال در محاسبه وزن چک می کنیم.)همچنین برای اینکه بتوانیم به طور کامل community ها را پیدا کنیم،باید community ها تعریف کنیم و مشخص کنیم دنبال چه community هایی با چه تعداد گره هستیم،و الگوریتم را به صورت بازگشتی روی دو مجموعه a و b اجرا کنیم،که در این پروژه این انجام نمی شود.

روند انجام پروژه

برای این پروژه از زبان پایتون استفاده شده است،با توجه به اینکه رسم نمودار در ان بسیار راحت بوده،و همچنین ساختمان داده های مناسبی برای پروژه دارد.همچنین به خاطر اینکه مقدار گره های موجود در تست ها زیاد بوده است،سعی شده است که از برنامه نویسی چند هسته ای استفاده شود،و برای همین دو ورژن،یکی با برنامه نویسی چند هسته ای ،و یکی با برنامه نویسی تک هسته ای تهیه شده است.

ساختمان داده ها

برای ذخیره سازی یال ها از dictionary استفاده شده است،که مانند یک hashmap می باشند و زمان دسترسی به هر یک از element های از مرتبه 1 است،و این برای ما بسیار مناسب است،با توجه به اینکه هر یک از یال ها،شناسه خودشان را دارند،که این شناسه نشان دهنده گره هایی است که این یال بین ان ها قرار دارد،و همچنین برای به دست اوردن تعداد دور های سه گره ای می توانیم از این نکته استفاده کنیم.همچنین برای پیاده سازی یال ها از کلاس استفاده شده است،که در ان وزن،cycle counter یا تعداد دور هایی که در ان مشارکت دارد،و وزن ان قرار دارد.برای ذخیره سازی گراف،ما از یک ماتریس 2 بعدی استفاده کردیم،که هر یک از ردیف های ان،یک dictionary می باشد ،که برای مثال ردیف i، گره هایی که گره i ام با ان ها در ارتباط است را ذخیره می کند.باز هم به خاطر اینکه هر یک از گره ها شناسه خودشان را دارند،به راحتی می توانیم از dictionary استفاده کنیم،و دلیل اینکه لایه اول ماتریس(ماتریسی که ردیف ها را ذخیره کرده است)نیز یک dictionary نمی باشد،این هست که هیچ یک از ردیف ها هیچ وقت حذف نمی شوند،و جایشان مشخص است،در حالی که گره ها های درون هر ردیف،با توجه به اینکه یال مربوطه موجود باشد یا نه،ممکن هست حذف شوند و برای همین ممکن است ترتیب به هم بخورد.همچنین ترتیب مشخصی در وصل بودن گره ها به هم وجود ندارد.توجه شود که اگر دو گره با هم ارتباط داشته باشند،ان گاه هر یک از ان ها در دریف مربوط به دیگری قرار دارند.برای پیاده سازی گره ها هم از کلاس استفاده می کنیم،که شامل اطلاعتی مانند شماره ان گره می شود.همچنین از ساختمان داده ارایه هم به عنوان stack در تابع bfs یا breath first search برای چک کردن اینکه ایا گراف همبند هست یا نه،و همچنین در تابع quicksort برای پیاده سازی ان به روش iterative، برای جلوگیری از stack overflow و با توجه به اینکه در پایتون recursion limit موجود هست ،استفاده شده است.در نهایت از queue هم برای برنامه نویسی چند هسته ای استفاده شده است، با توجه به اینکه این ساختمان داده shared vairable ها را مدیریت می کند.

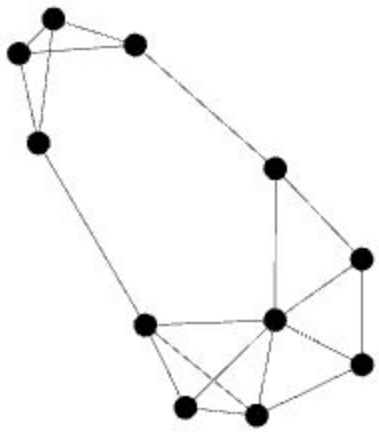
محاسبه ی امتیاز یال ها:

برای محاسبه امتیاز یال ها ما یک دفعه در ابتدای برنامه امتیاز یال ها را محاسبه می کنیم,با توجه به فرمولی که در پروژه تعریف شده است.در اینجا برای محاسبه تعداد دور ها ،برای به دست اوردن وزن یال ها ،از dictionary کمک می گیریم،و کافی هست ببینیم که

آیا بین گره های مربوط به یال(که در کلاس یال ذخیره شده است)،گره هایی هستند که به هر دو متصلند،و ما تعداد این گره ها را به دست می آوریم ،و همچنین این تعداد ها را درون کلاس یال ها،برای اصلاح وزن یال ها ذخیره می کنیم.بعد از حذف هر یالی ،ما وزن های یال های مربوط به ان را اصلاح می کنیم،که این یال ها شامل یال هایی هستند که به دو گره یال حذف شده متصل هستند،و حال با توجه به اینکه ایا یال حذف شده به کمک یال های مربوط دوری تشکیل داده بودند، اکنون تعداد ان دور ها را یکی کم کنیم،و درغیر این صورت،فقط دوباره مجبوریم از فرمول معرفی شده با توجه به به روز شدن درجه های گره ها استفاده کنیم.

فایل های تست:

با توجه به بزرگ بودن فایل های تست داده شده،و همچنین اینکه سرعت پایتون کم می باشد،از فایل های تست کوچک استفاده شده است،که شامل فایل های communitytest ها می باشند،که در آنها community ها با چشم قابل دیدن هستند،و بعد از انجام تست،می توان دید که در گراف حاصل شده،community ها از هم جدا شده اند.برای مثال:



و حال بعد از اجرای الگوریتم:

مثال بالا مربوط به فایل run bubblesort communitytest می باشد.