توضیح کد پروژه برش مینیمم گراف

Description Link: https://www.aparat.com/v/wVbm6

ابتدا گراف مورد نظر با دریافت ماتریس مجاورت گراف و با استفاده از دو ارایه رئوس و یال ها ساخته می شود.

رئوس گراف از نوع سوپر راس (SuperVertex) بوده که از کلاس Vertex ارث بری میکند و علاوه بر یک index که id آن است، ارایه ای از راس ها دارد که مشخص میکند این سوپر راس از ادغام چه راس هایی به وجود امده است.

یال های گراف از نوع سوپر یال (SuperVertex) بوده که علاوه بر یک راس ابتدایی و یک راس انتهایی شامل یک سوپرراس ابتدایی و یک سوپرراس انتهایی میشود و مشخص میکند راس ابتدایی در کدام سوپر راس و راس انتهایی در کدام سوپر راس قرار دارد.

پس از ساخته شدن گراف با استفاده از تابع getGraph() که ماتریس مجاورت گراف را به صورت یک رشته دریافت میکند و یک شی از نوع Graph برمیگرداند، فیلد استاتیک کلاس FxClass مقدار دهی شده و تابع drawMinCut() فراخوانی میشود تا برش مینیمم گراف رسم شود.

در تابع drawMinCut() علاوه بر دو مولفه گراف که پس از برش برخی یال ها ایجاد میشود، گراف اصلی نیز رسم میشود تا گراف ها با هم قابل مقایسه باشند.

برش مینیمم با استفاده از تابع getMinCutEdges() ساخته میشود. این تابع در صورتی که برش مینیمم قبلا ساخته شده باشد ان را باز میگرداند و در غیر اینصورت با استفاده از تابع applyKarger_Stein() الگوریتم Karger-Stein را روی گراف پیاده میکند و برش مینیمم را به دست می اورد.

در تابع applyKarger_Stein() متغیر radix مشخص کنند ضریب مورد استفاده در الگوریتم است که یک بار $\sqrt{2}$ و یک بار $\sqrt{3}$ در نظر گرفته شده است.

طبق این الگوریتم در صورتی که اندازه گراف یا همان تعداد رئوس آن n باشد، اگر $n > 4$ باشد تا زمانی که اندازه گراف به $n/radix$ برسد الگوریتم Karger اجرا میشود. در این الگوریتم در هر مرحله یک یال از گراف انتخاب شده و دو سر آن ادغام میشوند و یال هایی که به رئوس دو سر آن وصل بوده اند همگی به راس ادغامی جدید وصل میشند. همچنین یال هایی که یک سر آن ها یکی از راس های دو سر یال انتخابی باشد و سر دیگر آن ها راس دیگر یال انتخابی باشد باید وارد minCut شوند و در اخر جزو یال هایی هستند که حذف نمی شوند. در انتهای الگوریتم زمانی که تنها دو راس در گراف باشد تمامی یال های بین ان ها حذف میشود و گراف به دو مولفه تبدیل میشود.

زمانی که اندازه گراف به $n/radix$ رسید الگوریتم دو شاخه میشود و در هر شاخه یک یال به صورت رندم انتخاب میشود و دوباره الگوریتم Karger روی هرکدام اجرا میشود.

اگر $n \leq 4$ باشد تا انتها الگوریتم Karger اجرا میشود.

در تابع applyKarger_Stein() طبق الگوریتم بالا اگر $n > 4$ باشد الگوریتم Karger با استفاده از تابع applyKarger(double bound) اجرا میشود. این تابع عدد bound را به عنوان ورودی دریافت میکند که مشخص میکند الگوریتم تا چه زمانی تکرار شود. برای مثال زمانی که $n > 4$ است $n/_{radix}$ را به عنوان bound به تابع میدهیم تا طبق الگوریتم اندازه گراف کاهش یابد.

خروجی تابع applyKarger(double bound) یک گراف است که پس از بازگشتن الگوریتم Karger-Stein دو بار روی ان اجرا میشود.

در صورتی که $n \leq 4$ باشد تابع applyKarger(double bound) با bound = 2 فراخوانی میشود که اندازه گراف به ۲ برسد. پس از آن در صورتی که تعداد یال های موجود در ارایه minCut گراف برگردانده شده بیشتر از تعداد یال های موجود در ارایه minCutSaver باشد به این معنا است که در برش مینیمم گراف ساخته شده جدید تعداد یال های کمتری حذف شده اند و بنابراین برش مطلوب تری است و در نتیجه در minCutSaver جایگزین میشود.

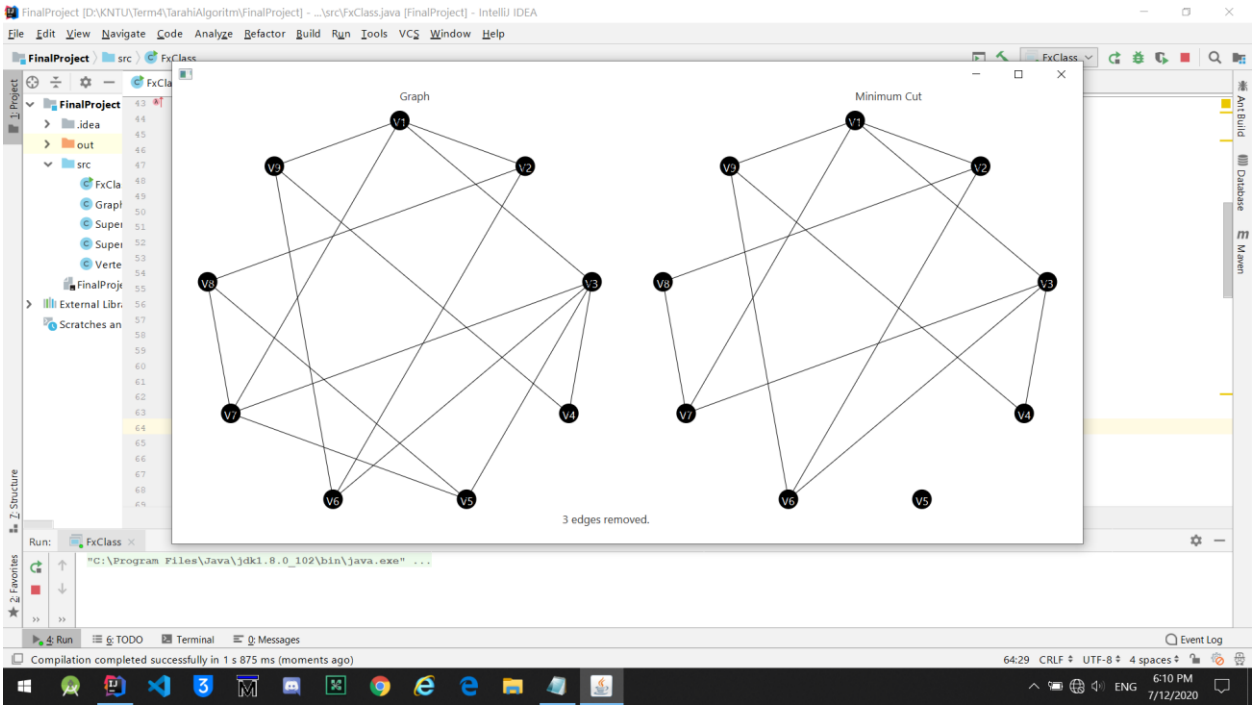
در انتها کوچیک ترین برش پیدا شده در ارایه minCutSaver قرار دارد که به عنوان مقدار بازگشتی تابع applyKarger_Stein() برمیگردد.

در تابع applyKarger(double bound) به منظور اینکه تغییری در گراف فعلی ایجاد نشود گراف جدیدی ساخته میشود و اطلاعات گراف فعلی به آن منتقل میشود. حلقه موجود در تابع مشخص میکند که الگوریتم تا زمانی که اندازه گراف به bound نرسیده تکرار شود. در هر مرحله طبق الگوریتم یک یال با استفاده از اندیسش در ارایه یال ها به صورت رندم انتخاب میشود و به minCutEdges گراف ساخته شده اضافه و از مجموعه یال های آن حذف میشود و سپس تابع merge() فراخوانی میشود تا راس ها و سایر یال ها ادغام شوند. این تابع سوپرراس های دو سر یال انتخابی را به عنوان ورودی دریافت میکند. در این تابع باز به منظور جلوگیری از ایجاد تغییر در گراف فعلی،سوپرراس جدیدی ساخته میشود و و تمام راس های موجود در سوپرراس های دو سر یال انتخابی در آن قرار داده میشود و دو سوپرراس از مجموعه رئوس حذف میشوند. بعد از آن تمام یال های گراف بررسی میشوند:

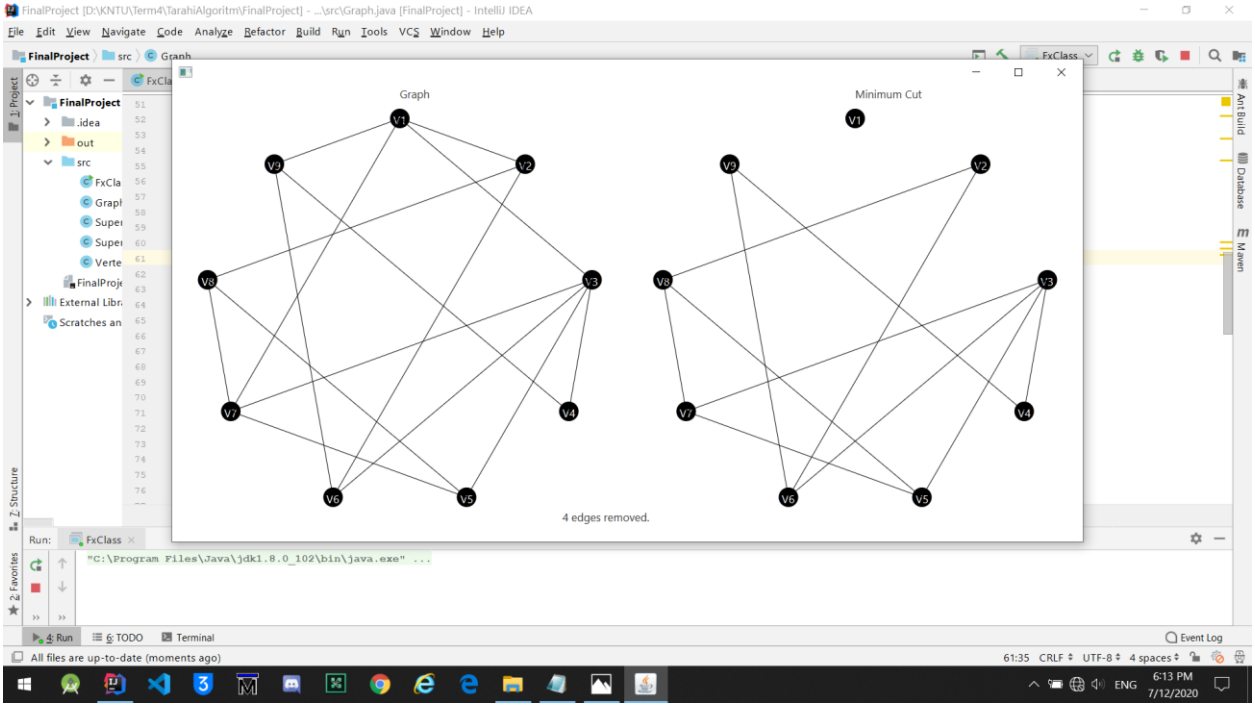
اگراز دو راس دو سر یک یال یکی در یک سوپرراس ورودی و دیگری در سوپرراس دیگر باشد باید وارد برش مینیمم شود و از مجموعه یال ها حذف شود.

اگر تنها یک سر یک یال در یکی از دو سوپرراس ورودی باشد، باید سوپرراس ابتدا یا انتهای آن یال برحسب شرایط تغییر کند.

پس اتمام عملیات یال هایی که از گراف حذف نمیشوند در minCutEdges قرار دارند و در کلاس FxClass و با استفاده از تابع getGraph() رسم میشوند.



radix = $\sqrt{2}$



radix = $\sqrt{3}$