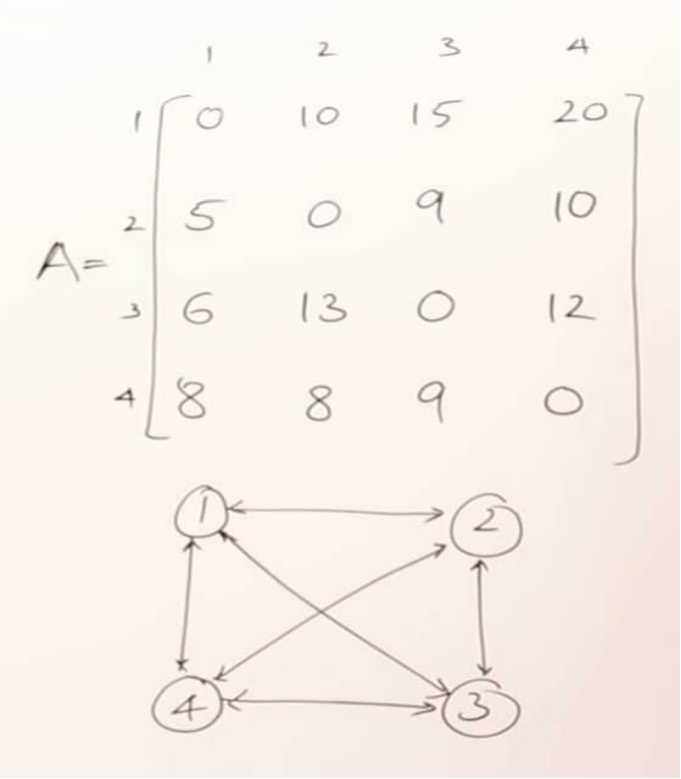
Genetic

پارسا محمدی

کد از لینک زیر قابل برسی و تغییر است و همچنین فایل آن هم در پوشه قرار دارد.

Code Link

- ابتدا کتابخانه ها را اضافه میکنیم
- بعد دیتا ها را لود می کنیم و نمایش می دهیم
- تابع distance فاصله بین نقاط را در فضای دو بعدی حساب می کند.
- تابع tsp_matrix ماتریس فواصل را به ما می دهد برای ساخت این ماتریس از تابع distance استفاده می کنیم. در این ماتریس فاصله هر دو شهر به صورت زیر ذخیره شده است:



- تابع make_population یک جامعه اولیه ایجاد می کند. کد این قسمت از ترکیب بخش make_solution و Find_neighbor دربخش تپه نوردی گرفته شده است. این کار برای ساخت جامعه بهترین راه نیست ولی من این را انتخاب کردم.
- تابع cost هم با استفاده فواصل ذخیر شده در در ماتریس فواصل (به اسم tsp در این کد) مجموع فواصل تا نود هدف را می یابد

- تابع top_cost لیستی از solution ها یا همان population را می گیرد و کاست آن ها را حساب می کند و بهترین آن ها را می یابد، هر مسیر (solution) را با کاست آن تبدیل به یک لیست می کند و همه مسیر ها و کاست آن ها که لیست شدند را هم لیست می کند و خروجی می دهد. ورودی n هم تعداد بهترین کاست خواهی است که می خواهیم. مثلا برای که جامعه 20 تایی اگر n 5 باشد این تابع 5 تا از بهترین کاست ها را خروجی می دهد.
- تابع cross_over یک population را می گیرد مسیر ها را از کاست ها جدا می کند و اگر random.random() از prob کمتر بود بخش هایی از آن را به صورت رندم انتخاب می کند و با مسیر بعدی عوض می کند. کاست مسیر های ایجاد شده جدید را حساب می کند و مانند تابع قلبی کاست آن ها را حساب می کند و به صورت لیست خروجی می دهد
- تابع mutation هم مانند cross_over مسیر ها را از کاست ها جدا می کند و و اگر random.random() از prob کمتر بود المان هایی از آن را به صورت رندم انتخاب می کند و مقدار آن ها را از کل تعداد شهر ها کم می کند تا جامعه و ژن جدید ایجاد بشود
کاست نسل جدید محاسبه شده و با مسیر آن لیست می شود و به صورت لیست خروجی می دهد.
- در تابع ژنتیک هم الگوریتم ژنتیک ایجاد می شود. ابتدا یک جامعه ایجاد می شود بهترین کاست های آنها انتخاب می شوند و در حلقه روی آن ها عملیات cross_over و mutation انجام می شود و مقدار کاست در هر مرحله ذخیره می شود برای رسم نمودار.

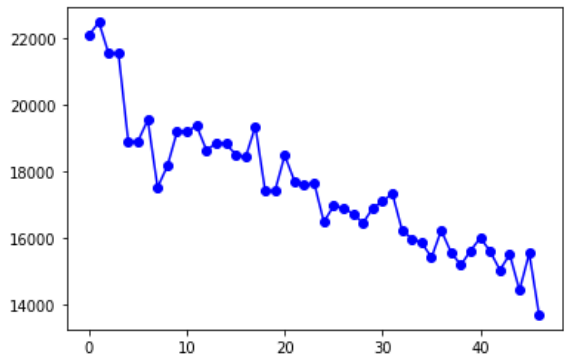
نتایج:

دیتای جیبوتی بعد از 47 بار تکرار

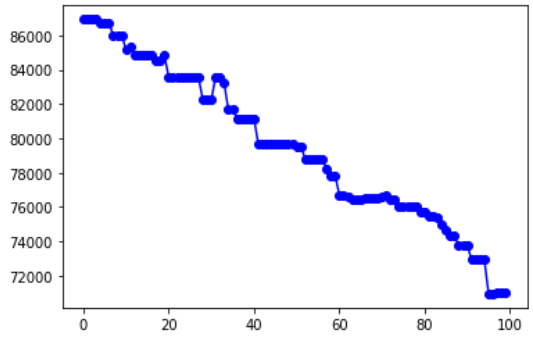
روند نزولی کاست در هر مرحله قابل تشخیص است.

کاست نهایی در این حالت از SA بیشتر است علت آن این است که تعداد تکرار ها برای SA 5000 بار بود ولی در این حالت 47 بار. مشخص است که شیب همگرایی الگوریتم ژنتیک بسیار بیشتر است.

```
ittiration : 37
ittiration : 38
ittiration : 39
ittiration : 40
ittiration : 41
ittiration : 42
ittiration : 43
ittiration : 44
ittiration : 45
ittiration : 46
13695.55278883902
```



```
ittiration : 87
ittiration : 88
ittiration : 89
ittiration : 90
ittiration : 91
ittiration : 92
ittiration : 93
ittiration : 94
ittiration : 95
ittiration : 96
ittiration : 97
ittiration : 98
ittiration : 99
71034.74417905803
```



نتایج دیتای قطر به صورت رو به رو است. با توجه به اینکه فقط 100 تکرار انجام شده است به نتایج خیلی بهتری نسب به الگوریتم SA رسیدیم

کاست بعد از 100 تکرار 71034 می باشد.

در نهایت:

الگوریتم Hill climbing الگورتیمی است ک ممکن است هیچگاه به گلوبال اپتیما میل نکند و اکثرا در لوکال اپتیما گیر می کند.

الگورتیم Simulated annealing مشکل الگورتیم تپه نوردی در آن حل شده است و به گلوبال اپتیما میل می کند. ولی سرعت میل کردن آن کم است.

الگورتیم Genetic هم به گلوبال اپیتما میل می کند و سرعت آن میل کردن آن به گلوبال اپتیما بیشتر است.

END 😊