به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

# درس حسابگری زیستی

# تمرین اول

| | |
|---|---|
| نام و نام خانوادگی | سارا رستمی |
| شماره دانشجویی | ۸۱۰۱۰۰۳۵۵ |
| تاریخ ارسال گزارش | ۱۴۰۲.۰۱.۱۶ |

## فهرست

# شکل‌ها

# پاسخ ۱.

## ۱-۱. مدلسازی

مسئله‌ی بیان‌شده، در واقع حالت خاصی از مسئله‌ی مجموعه مستقل ماکسیمم[۱] می‌باشد. با در نظر گرفتن این نکته، fitness function را برای حل مسئله‌ی مجموعه مستقل ماکسیمم مطابق پژوهش Back و Khuri [۱]، تعریف کردیم. فرمول (۱) تابع fitness ما را نشان می‌دهد، که در آن n برابر با تعداد رأس‌های گراف بوده و $x_i$ یکی از مقادیر ۰ یا ۱ را گرفته که به ترتیب، انتخاب نشدن یا شدن رأس iام را نشان می‌دهد. علاوه بر این، $e_{ij}$ عدم وجود یا وجود یال بین رأس‌های i و j را به ترتیب با گرفتن مقادیر ۰ و یا ۱ نشان می‌دهد. و از این رو، یکی از شرایط خاتمه الگوریتم (stopping criteria) را رسیدن به حداقل یک جواب با اندازه m، قرار دادیم. بنابراین، الگوریتم ممکن است سه نوع جواب داشته باشد: ۱- الگوریتم نتواند تا تعداد max iterations، جواب با اندازه m پیدا کند. که در این صورت، نزدیکترین جواب پیدا شده برگردانده می‌شود ۲- الگوریتم حداقل یک جواب با اندازه m پیدا کرده و اجرای الگوریتم خاتمه می‌یابد. که در این صورت، همین جواب برگردانده می‌شود ۳- الگوریتم حداقل یک جواب با اندازه بزرگتر از m پیدا کرده و خاتمه می‌یابد. که در این صورت، m رأس به صورت تصادفی از بهترین جواب یافته شده انتخاب شده و برگردانده می‌شود.

$$f(\vec{x}) = \sum_{i=1}^{n} \left( x_i - n x_i \sum_{j=i}^{n} x_j \, e_{ij} \right) \qquad (۱)$$

## ۱-۲. ارزیابی

هدف تنظیم مناسب پارامترها به گونه‌ایست که با صرف کمترین منابع در سریعترین زمان جواب موردنظر پیدا شود. از آنجایی که نمونه گراف ورودی (آورده شده در صورت سوال) ، بسیار کوچک بود، بدون انجام هیچگونه تنظیم پارامتری، طی تنها یک iteration به جواب می‌رسیدیم. به طور مشخص، در ابتدا پارامترها را همانند پژوهش Back و Khuri [۱]، قرار دادیم که عبارتند از:

Population size = 50 , mutation rate = 1/n , crossover rate = 0.6 ,
selection type = proportional selection , crossover type = two-point crossover

[۱] maximum independent set

لازم به ذکر است که یکی از روش‌های تنظیم پارامتر در الگوریتم‌های تکاملی بر اساس Analogy است[۲]. به این معنی که، پارامترهایی که برای یک مسئله تنظیم شده‌اند، برای مسئله‌ای با ویژگی‌های مشابه، احتمالاً عملکرد خوبی خواهند داشت. با درنظرگرفتن این نکته، و شباهت بدیهی مسئله‌ی ما به مسئله‌ی مجموعه مستقل ماکسیمال، پارامترهای اولیه را بر اساس پژوهش ذکر شده را تنظیم کردیم.

علاوه بر این، برای اینکه پارامترهای مناسب برای مسئله مجموعه مستقل با اندازه k را تحقیق کنیم، آزمایشی زیر را طراحی کردیم:

۱- تعدادی گراف تصادفی با تعداد رئوس کم (چرا که اندازه نمونه گراف ورودی کوچک بود) و تراکم یال نسبتاً زیاد (چرا که نسبت تعداد یال‌ها به حداکثر تعداد یال ممکن در نمونه گراف داده‌شده برابر با 0.66 بود) ، تولید کردیم.

۲- سپس مسئله‌ی تنظیم پارامترهای الگوریتم ژنتیک را به صورت یک مسئله‌ی بهینه‌سازی دو هدفه مدل کردیم.که در آن هدف کمینه کردن میانگین تعداد تکرارها تا رسیدن به جواب بین همه‌ی گراف‌های تصادفی تولید شده و بیشینه کردن میانگین اندازه‌ی مجموعه مستقل‌ها بین گراف‌های مذکور می‌باشد.

۳- در نهایت به کمک روش بهینه‌سازی Bayesian و با استفاده از الگوریتم Tree-structured Parzen Estimator (TPE) به جستجوی فضای حالات پارامترها پرداختیم. شکل ۱ این فضای حالات را نشان می‌دهد.

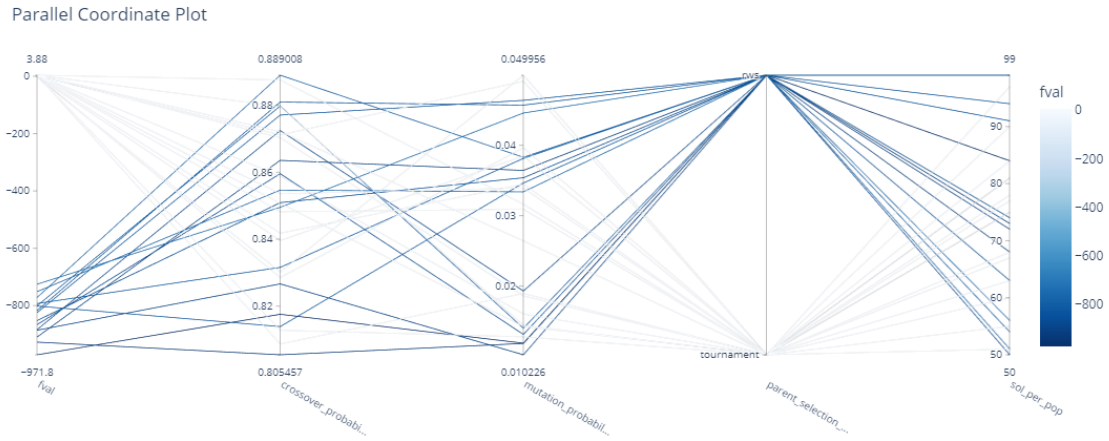
```
params = {
    "sol_per_pop": trial.suggest_int("sol_per_pop", 50, 100),
    "num_parents_mating":2,
    "crossover_probability":trial.suggest_float("crossover_probability",0.8,0.9,log=False),
    "mutation_probability":trial.suggest_float("mutation_probability",0.01,0.05,log=False),
    "gene_space":[0,1],
    "parent_selection_type":trial.suggest_categorical("parent_selection_type", ["rws","tournament"]),
    "num_generations":100
}
```

شکل ۱- فضای حالات پارامترها

لازم به ذکر است فضای حالات، مطابق پیشنهاد ChatGPT در مورد مقادیر معمول برای پارامترهای شکل ۱، تنظیم شده‌است. شکل ۲ خلاصه‌ی نتایج آزمایش مربوط به parameter tuning را برای کمینه‌سازی مقدار میانگین تعداد iteration‌ها و بیشینه‌سازی میانگین fitness value ها را نشان می‌دهد. مطابق این شکل، بهترین پارامترهای بدست آمده عبارتند از:

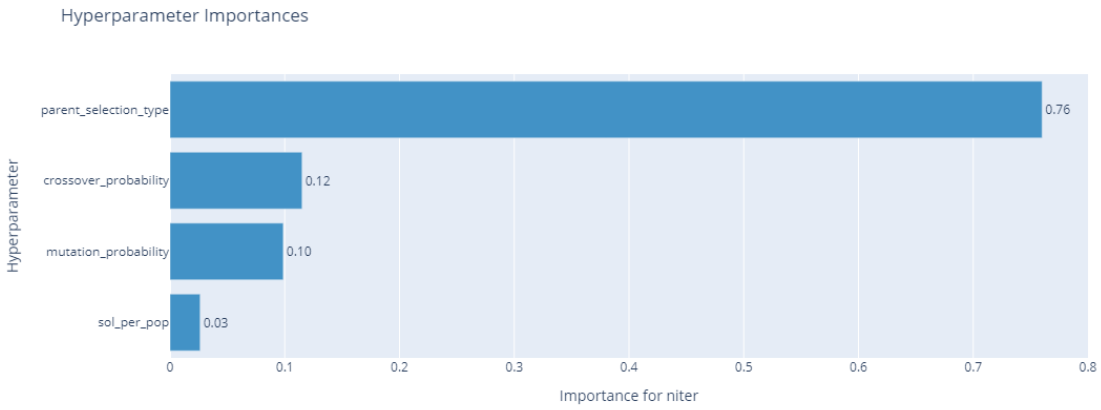
Population size = 63 , mutation rate = 0.0264, crossover rate = 0.8721,

parent selection type = tournament , crossover type = two-point crossover

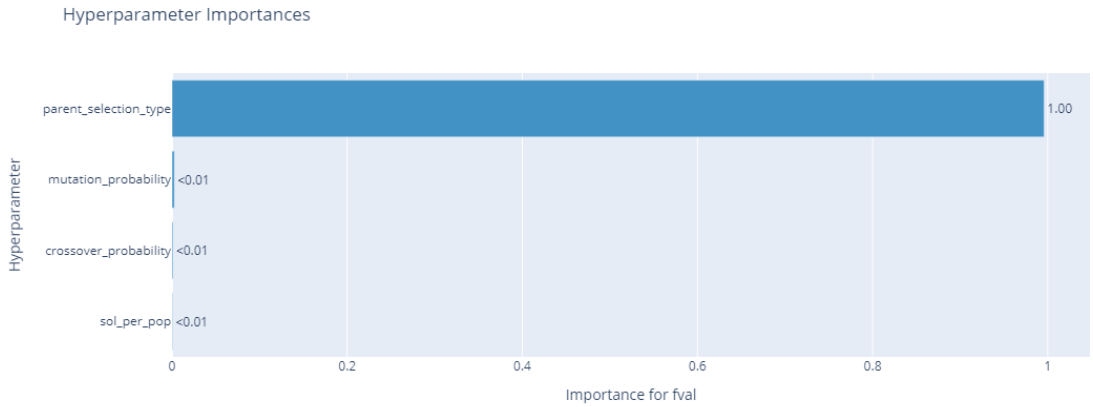



شکل ۲- نمودار **parallel coordinate** برای تنظیم پارامترها

در شکل‌های ۳ و ۴، میزان اهمیت هر یک از پارامترها برای رسیدن به دو هدف ما (بیشنیه کردن مقدار fitness function و کمینه‌کردن تعداد iterationها) نشان داده شده است. مطابق این شکل‌ها، استراتژی انتخاب والد، بیشترین اهمیت را برای بهینه‌سازی هر دو هدف دارد.



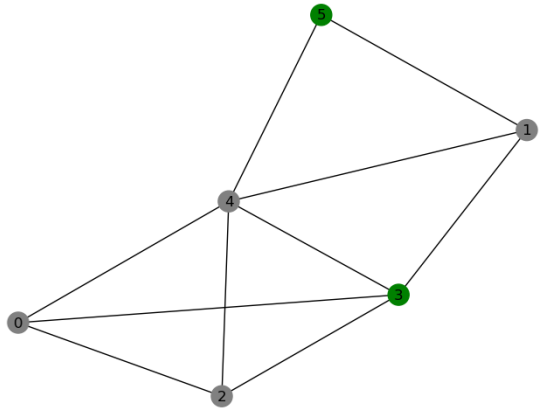

شکل ۳- نمودار اهمیت پارامترها در کمینه کردن تعداد **itertaion**ها




شکل ۴- نمودار اهمیت پارامترها در بیشینه کردن مقدار **fitness function**

نتایج و جمع‌بندی:

پس از بدست آوردن بهترین پارامترها با استفاده از parameter tuning توضیح داده شده در بالا، الگوریتم را با پارامترهای مذکور برای گراف نمونه ورودی، اجرا کردیم. به علت کوچکی نمونه ورودی، نتایج بدست آمده بعد از تنظیم پارامتر تفاوتی با نتایج با مقادیر ست شده طبق مقاله [۱] نداشت و الگوریتم طی یک iteration به جواب رسید. جواب پیدا شده توسط الگوریتم را در شکل ۵ مشاهده می‌کنید.




شکل ۵- پاسخ پیدا شده به ازای نمونه گراف ورودی سوال ۱

همانطور که در شکل ۵ مشاهده می‌کنید، رأس‌های ۳ و ۵ به عنوان پاسخ انتخاب شدند که شماره این رئوس را در دو سطر در فایل Q1_output.txt نوشتیم.

# پاسخ ۲. مسئله ZOE

## ۲-۱. مدلسازی

مسئله‌یZOE را به این صورت مدل‌سازی می‌کنیم که هر کروموزوم برداری n عنصری را نمایندگی می‌کند که در آن هر عنصر می‌تواند مقدار ۰ و یا ۱ بگیرد. سپس تابع fitness را به این صورت تعریف می‌کنیم: حاصل جمع درایه‌های مساوی ۱ در بردار حاصل از ضرب ماتریس A در ماتریس x

که سعی داریم آن را بیشینه کنیم. با تعریف چنین تابع برازشی، سعی بر این است که تعداد یک‌های بردار خروجی ماکسیمم شود.

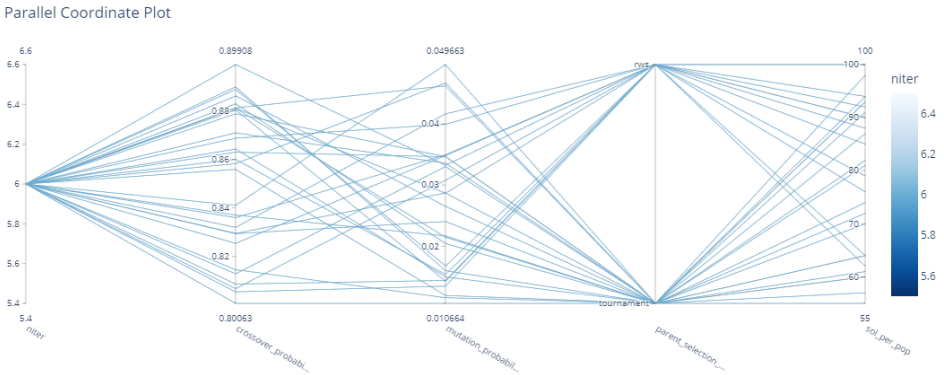
## ۲-۲. ارزیابی

برای ارزیابی مشابه آزمایش توضیح داده‌شده در قسمت ۲-۱ سوال قبل عمل کردیم. نمودارها و پارامترهای بدست آمده با parameter tuning را در ادامه آورده و توضیح می‌دهیم.

دقت شود که برای نمونه ماتریس ورودی الگوریتم با هر پارامتری (در بازه‌ی مشخص شده‌ای که تعیین کردیم) همواره جواب را پیدا می‌کند و مقدار fitness function بیشینه خود را، که در اینجا برابر با m می‌باشد، بدست می‌آورد. پس هدف از تنظیم پارامترها در اینجا، تنها کمینه کردن تعداد iterationهای اجرای الگوریتم می‌باشد.

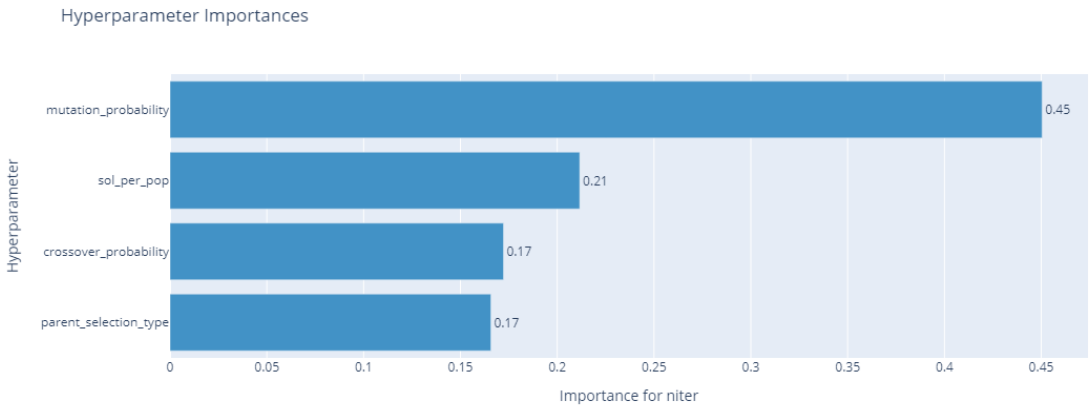
شکل ۶ خلاصه‌ی نتایج آزمایش مربوط به parameter tuning را برای کمینه‌سازی مقدار میانگین تعداد iterationها را نشان می‌دهد. مطابق این شکل، بهترین پارامترهای بدست آمده عبارتند از:

Population size = 85 , mutation rate = 0.0132, crossover rate = 0.8442,
parent selection type = tournament , crossover type = two-point crossover




شکل ۶- نمودار **parallel coordinate** برای تنظیم پارامترها

طبق شکل ۷، مهمترین پارامتر در کمینه کردن تعداد iterationهای الگوریتم، پارامتر mutation probability می‌باشد.




شکل ۷- نمودار اهمیت پارامترها در کمینه کردن تعداد **itertaion**ها

نتایج و جمع‌بندی:

پس از بدست آوردن بهترین پارامترها با استفاده از parameter tuning توضیح داده شده در بالا، الگوریتم را با پارامترهای مذکور برای ماتریس نمونه ورودی، اجرا کردیم. به علت کوچکی نمونه ورودی، نتایج بدست آمده بعد از تنظیم پارامتر تفاوتی با نتایج با مقادیر ست شده اولیه نداشت و الگوریتم طی یک iteration به جواب رسید. جواب پیدا شده توسط الگوریتم برای بردار x برابر با [1, 1, 0, 1, 0] می‌باشد.

مقدار بدست آمده برای هر یک از درایه‌های بردار x را در یک سطر جداگانه در فایل Q2_output.txt نوشتیم.

# پیاده‌سازی

تمام کد به زبان Python 3 نوشته شده است. برای پیاده‌سازی الگوریتم ژنتیک، از کتابخانه‌ی PyGAD و برای تنظیم پارامترهای آن از کتابخانه‌ی optuna استفاده کردیم.

کد تمرین در ۳ نوت‌بوک پایتون (.ipynb) نوشته شده‌است. فایل Q1_1.ipynb، یک فایل txt. به عنوان ورودی دریافت کرده و سپس مسئله سوال ۱ را به کمک پارامترهای تنظیم شده، حل کرده و خروجی را در فایل Q1_output.txt می‌نویسد. فایل Q1_2.ipynb، حاوی کد مربوط به parameter tuning الگوریتم ژنتیک برای سوال ۱ می‌باشد. فایل Q2_1.ipynb و Q2_2.ipynb برای سوال ۲ همانند فایل‌های سوال ۱ سازماندهی شده‌اند.

نکته قابل توجه دیگر این است که در الگوریتم ژنتیک پیاده‌سازی شده در کتابخانه‌ی PyGAD اندازه جمعیت در نسل‌های مختلف یکسان است و متغیر نمی‌باشد. از این رو تعداد اعضای population در حین اجرای الگوریتم ثابت می‌ماند.

# مراجع


1- Back, T., & Khuri, S., "An evolutionary heuristic for the maximum independent set problem", In IEEE World Congress on Computational Intelligence, pp. 531-535,1994.

2- Eiben, Á. E., Hinterding, R., & Michalewicz, Z. "Parameter control in evolutionary algorithms", IEEE Transactions on evolutionary computation, pp. 124-141,1999.