

**درس رایانش تکاملی مبتنی بر جمعیت**

تمرین چهارم

**پیاده سازی و بررسی الگوریتم ژنتیک با انتخاب Tournament و کد گذاری اعداد حقیقی**

نام دانشجو:

نوید حاجی زاده

4012366057

نام استاد درس:

دکتر مجتبی روحانی

پاییز 1402

**چکیده:**

در این تمرین همانند تمرین سری قبل، خواسته شده که الگوریتم ژنتیک پیاده سازی شود و بر روی چندین تابع تست ارزیابی شود. تنها تفاوت این تمرین با تمرین سری قبل در این است که به جای کد گذاری باینری، از کد گذاری اعداد حقیقی و برای انجام عمل انتخاب (Select) از روش Tournament استفاده شود.

لازم به ذکر است که توابع استفاده شده در این تمرین، همان ده تابع تمرین قبل می‌باشند که آنها را با F1تا F10 نمایش داده ایم. در این صورت الگوریتم ژنتیک را بر روی این توابع اعمال کرده و بررسی می‌کنیم که تغییر پارامترهای الگوریتم تعداد جمعیت، احتمال تقاطع و احتمال جهش به چه میزان بر روی نتیجه اثر می‌گذارد.

برای پیاده‌سازی الگوریتم ژنتیک، می‌دانیم که هر عضوی از جمعیت، خود شامل N عدد است. هر کدام از این N عدد به صورت حقیقی هستند. برای انجام تقاطع احتمالی دو والد، از روش تقاطع حسابی استفاده شده است. همچنین برای عمل جهش، فرزند انتخاب شده با بردار استاندارد تصادفی به میانگین 0 و واریانس قابل تنظیم σ جمع می‌شود.

**شرح نتایج:**

همانند تمرین پیشین، در ابتدا می‌بایست با استفاده از یکی از توابع گفته شده، مقادیر بهینه هر کدام از پارامترها را تعیین کنیم. ما برای مراحل بعد، از این مقادیر استفاده خواهیم کرد. مجددا مانند تمرین قبل از تابع هدف F5 برای این کار استفاده می‌کنیم، و مقدار N را برابر 10 در نظر می‌گیریم.

ابتدا با در نظر گرفتن مقدار ثابت احتمال تقاطع 0.9 و احتمال جهش 0.1، تابع فوق را برای مقادیر مختلف جمعیت بررسی می‌کنیم:

|  |  |
| --- | --- |
| population\_size | objective\_function\_score |
| 50 | 9.56 |
| 100 | 8.67 |
| 250 | 6.7 |
| 500 | 7.29 |

آنچه از جدول بالا می‌توان نتیجه گرفت این است که میزان جمعیت برای مینیمم کردن تابع ارزیاب ما برابر با 250 است. حال با تنظیم مقدار جمعیت و فرض اینکه مقدار ثابتی برای احتمال جهش وجود دارد، آن را برابر 0.1 در نظر گرفته و لذا تابع را برای مقادیر مختلف احتمال تقاطع بررسی می‌کنیم:

|  |  |
| --- | --- |
| p\_crossover | objective\_function\_score |
| 0.2 | 10.3 |
| 0.5 | 8.58 |
| 0.7 | 9.62 |
| 0.9 | 8.74 |

بعد از محاسبه تابع هدف، در می‌یابیم که بهترین مقدار برای احتمال تقاطع، میزان 50 درصد است.

تا به اینجا مقدار بهینه جمعیت و احتمال تقاطع را تعیین کرده ایم، حال تابع را برای مقادیر مختلف احتمال جهش بررسی می‌کنیم:

|  |  |
| --- | --- |
| p\_mutation | objective\_function\_score |
| 0.01 | 9.17 |
| 0.05 | 8.72 |
| 0.1 | 7.6 |
| 0.2 | 9.49 |
| 0.5 | 7.94 |

مجددا با ارزیابی تابع هدف، بهینه ترین جواب برای احتمال جهش، برابر با 10 درصد است.

با تعیین سه پارامتر قبلی حال به سراغ تعیین مقدار برای حالات مختلف تعداد فرزندان شرکت کننده در هر Tournament یعنی k رفته و آن را بررسی می‌کنیم:

|  |  |
| --- | --- |
| k | objective\_function\_score |
| 1 | 61.27 |
| 3 | 8.99 |
| 5 | 9.31 |
| 7 | 9.97 |

به نظر می‌آید که مقدار مناسب برای تعداد فرزندان شرکت کننده در هر Tournament برابر 3 ‌باشد.

حال که به مقادیر بهینه برای پارامترهای بخش‌های قبل رسیدیم، وقت آن است تا مقدار مناسب را برای پارامتر جهش σ تعیین کنیم:

|  |  |
| --- | --- |
| σ | objective\_function\_score |
| 0.01 | 19.3 |
| 0.05 | 9.57 |
| 0.1 | 7.31 |
| 0.2 | 8.28 |
| 0.5 | 8.72 |

با محاسبه تابع هدف، مقدار مناسب برای پارامتر جهش σ برابر 0.1 می‌باشد.

با مشخص شدن مقادیر مناسب برای پارامترها، حال به بررسی نتایج حاصل از اعمال الگوریتم ژنتیک بر روی هرکدام از توابع تست به ازاء 50 30, N=10, می‌پردازیم. دقت شود که هر بار 10 دفعه اجرا تکرار شده است و نهایتا جواب نهایی میانگین جوابهای محاسبه شده، می‌باشد. همچنین در نظر داشته باشید که آنچه محاسبه شده است، برای تعداد تکرار (Iteration) برابر 1000 است.

|  |  |  |  |
| --- | --- | --- | --- |
|  | N=10 | N=30 | N=50 |
| F1 | 0.0009 | 0.072 | 0.507 |
| F2 | 0.058 | 0.324 | 2.231 |
| F3 | 0.0097 | 60.272 | 1782.247 |
| F4 | 0.032 | 8.248 | 10.324 |
| F5 | 7.688 | 33.285 | 246.729 |
| F6 | 0.0007 | 0.0499 | 29.469 |
| F7 | 1.259 | 8.674 | 17.239 |
| F8 | -2340.051 | -4728.528 | -5480.22 |
| F9 | 0.0005 | 0.0324 | 21.533 |
| F10 | 1.844 | 2.392 | 3.525 |

**نتیجه گیری:**

اگر نیک به نتایج بنگریم، در می‌یابیم که با افزایش N ، رسیدن به جواب بهینه سخت‌تر شده است. علت آن است که الگوریتم باید تعداد بیشتری عدد را بهینه‌سازی کند.

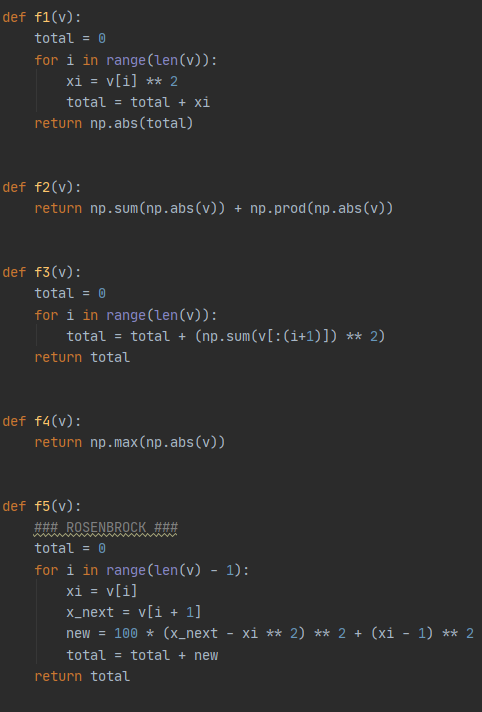
در صورت مقایسه نتایج این تمرین با تمرین سوم، به وضوح مشخص است که مقادیر تابع برای این تمرین به مراتب از تمرین پیشین بهتر هستند. پس می‌توانیم نتیجه بگیریم که در توابع مورد بررسی، استفاده از روش انتخاب Tournament به همراه کد گذاری اعداد حقیقی، نتایج بسیار بهینه‌تری نسبت به روش انتخاب Wheel Roulette به همراه کد گذاری باینری می‌دهد.

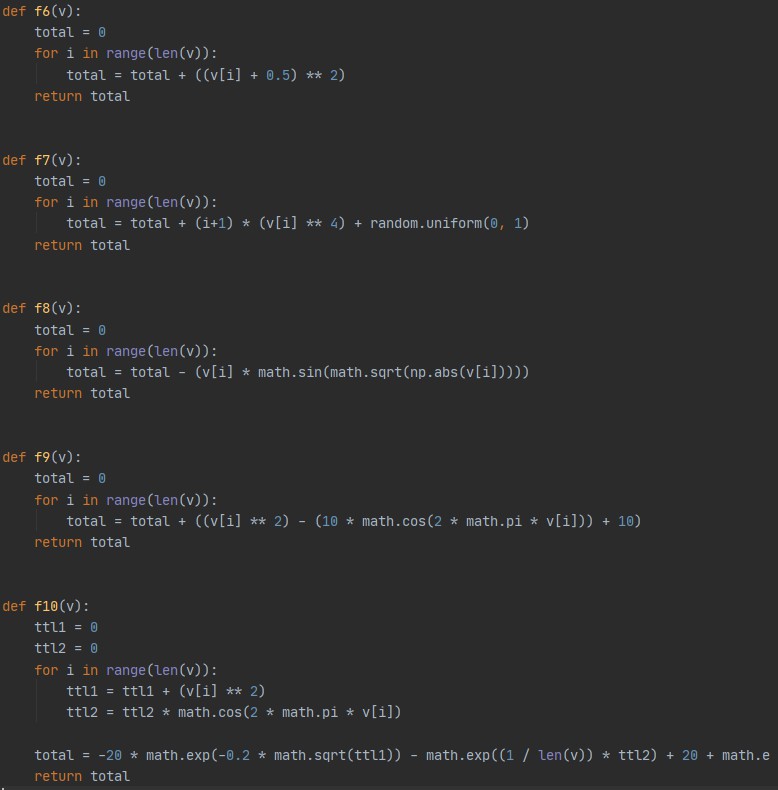
علت می‌تواند در این باشد که در روش انتخابی Tournament، شانس بیشتری برای انتخاب عضوهای ضعیف‌تر در جامعه فعلی وجود دارد، زیرا در این روش انتخاب مقادیر مطلق تابع هدف مهم نیستند و به تعبیری میزان اختلاف بین تابع هدف عضوهای جامعه تاثیری در شانس برنده شدن آنان ندارد. این امر باعث می‌شود تا اکتشاف در جهت پیدا کردن جواب بهینه به نحو بهتری انجام شود.

همچنین در هنگام انجام عمل جهش بر روی اعضا دارای کدگزاری حقیقی، از تغییرات ناگهانی فرد پس از جهش جلوگیری می‌شود، که این امر می‌تواند در گرفتن نتایج بهتر نقش مهمی داشته باشد.

**ضمیمه کد :**

**پیاده سازی توابع هدف:**





**پیاده سازی الگوریتم :GA**

