دانشگاه شهید بهشتی

دانشکده مهندسی و علوم کامپیوتر

گزارش تمرین دوم درس یادگیری ماشین

# پیاده سازی الگوریتم ژنتیک کلاسیک در پایتون و بهبود آن با پذیرش حریصانه ی ترکیب، جهش، و نسل جدید

نگارش

**سهیل ضیائی قهنویه**

استاد

**دکتر احمد علی آبین**

اردیبهشت ۱۴۰۰

## چکیده

الگوریتم ژنتیک یکی از الگوریتم های محاسباتی تکاملی است که با الهام از اصل "بقای متناسب ترین" با تکرار عملیات های انتخاب، ترکیب و جهش راه حل های نسبتاً قابل قبولی برای مسائل NP-Complete ارائه می دهد. در این پروژه پس از پیاده سازی الگوریتم ژنتیک کلاسیک، بهبودهایی برای آن پیشنهاد شده است و میزان تأثیر این بهبودها با استفاده از توابع ارزیابی Unimodal و Multimodal مقایسه و بررسی شده است.



## کلیدواژه‌ها

محاسبات تکاملی، الگوریتم ژنتیک، تابع هدف، تابع تناسب، کروموزوم، ژن، انتخاب، ترکیب، جهش، توابع ارزیابی، پایتون

# فهرست نوشتار


چکیده ..... ۲

کلیدواژه‌ها ..... ۲

فهرست نوشتار ..... ۳

بخش اول: مقدمه ..... ۵

محاسبات تکاملی و الگوریتم ژنتیک ..... ۵

معرفی الگوریتم ژنتیک کلاسیک ..... ۶

انتخاب ..... ۶

ترکیب ..... ۶

جهش ..... ۷

ساختار گزارش ..... ۷

بخش دوم: الگوریتم های پیشنهادی ..... ۸

بهبود اول – پذیرش حریصانه ترکیب ..... ۸

بهبود دوم – پذیرش حریصانه نسل جدید ..... ۸

بهبود سوم – پذیرش حریصانه جهش ..... ۸

بهبود چهارم – ترکیب بهبودهای قبل ..... ۸

بخش سوم: روش ارزیابی ..... ۹

توابع ارزیابی ..... ۹

دنباله داده های تصادفی یکسان ..... ۹

ارزیابی میزان کاهش خطا در هر نسل ..... ۱۰
ارزیابی میزان کاهش خطای نهایی ..... ۱۰
بخش چهارم: نحوه پیاده سازی ..... ۱۲
زبان و پارادایم برنامه نویسی ..... ۱۲
ساختار پروژه ..... ۱۲
کد گذاری کروموزوم ها ..... ۱۲
تابع هدف و تابع تناسب ..... ۱۲
بخش پنجم: ارائه نتایج ..... ۱۴
ارزیابی میزان کاهش خطا در هر نسل ..... ۱۴
ارزیابی میزان کاهش خطای نهایی ..... ۱۶
مراجع ..... ۱۸

# بخش اول: مقدمه

## محاسبات تکاملی[۱] و الگوریتم ژنتیک[۲]

محاسبات تکاملی شامل تکنیک های جستجوی احتمالی الهام گرفته از اصل "بقای متناسب ترین" نظریه تکامل طبیعی داروین و مکانیسم های ژنتیک طبیعی است. این تکنیک ها بر روی جمعیتی از "موجودات" مصنوعی کار می کنند، که در مبارزه برای زندگی که در آن سالم ترین افراد زنده می مانند و اجازه تولید مثل دارند، رقابت می کنند. موجودات جدید با استفاده از تکه های اعضای نسل قبلی ایجاد می شوند و گاهی اوقات، موجودات به طور تصادفی تغییر می کنند.

الگوریتم های تکاملی تصادفی هستند، اما به هیچ وجه پیاده روی های تصادفی (random walk) ساده نیستند. آنها فضای جستجو را با استفاده از اطلاعات تاریخی پیمایش می کنند تا حدس بزنند کجا می توانند به جستجوی نقاط بهتر بپردازند. الگوریتم هایی از این نوع برای مسائلی که معلوم نیست هیچ راه حل کارآمدی وجود دارد، بهترین حالت را دارند. الگوریتم های خاص منظوره، مثلا الگوریتم هایی که دارای مقدار مشخصی از دانش حوزه مسئله هستند که به سختی رمزگذاری شده اند، معمولاً از الگوریتم های ژنتیک پیشی می گیرند. با این حال، برای بسیاری از مسائل NP-Complete، الگوریتم های ژنتیک از بهترین استراتژی های شناخته شده هستند. این همان چیزی است که الگوریتم های ژنتیک به ویژه در آن مهارت دارند: یافتن راه حل هایی که همه موارد دیگر به نتیجه نرسد. تا به امروز، بسیاری از الگوریتم های ژنتیک برای حل مسائل متنوعی از زمان بندی[۳] بازی ها[۴] استنباط دستوری[۵] بسته بندی[۶] فروشنده دوره گرد[۷] و بسیاری دیگر استفاده کرده اند [1] .

---

۱ Evelutionary Computation

۲ Genetic Algorithm

۳ time-tabling and job-shop scheduling

۴ game playing

۵ grammatical inference

۶ bin packing

۷ traveling salesman

## معرفی الگوریتم ژنتیک کلاسیک

عملکرد الگوریتم ژنتیک کلاسیک بسیار ساده است. در هر مرحله نسلی از کروموزوم[۸]ها که راه حل های بالقوه ی تابع هدف[۹] هستند را نگهداری می کند. این کروموزوم ها رشته هایی هستند که خصوصیات راه حل بالقوه را به صورت کد شده شامل می شوند. به هر یک از اجزای کروموزوم ژن[۱۰]گفته می شود. در الگوریتم ژنتیک کلاسیک، از کد گذاری باینری استفاده می شود. این الگوریتم دارای سه عملگر است که در بخشهای بعدی مورد بحث قرار می گیرند: انتخاب، ترکیب و جهش [1].

## انتخاب[۱۱]

انتخاب فرآیند ارزیابی میزان کیفیت و تناسب کروموزوم های یک نسل و انتخاب کروموزوم های مجاز به تولید مثل است. در طبیعت، تناسب توسط توانایی موجودات برای غلبه بر خطرات بلوغ و تولید مثل تعیین می شود. در محیط مصنوعی، برای ارزیابی تناسب کروموزوم ها به تابع تناسب[۱۲]نیاز است [1]. در الگوریتم ژنتیک کلاسیک، در هر مرحله از میان نسل فعلی دو کروموزوم انتخاب می شود تا در نسل جدید از آن ها استفاده شود. در این انتخاب کروموزوم های با تناسب بالاتر شانس بیشتری برای انتخاب دارند.

## ترکیب[۱۳]

در عملیات ترکیب، بعد از انتخاب دو کروموزوم والد (در عملیات انتخاب) با مبادله قسمتهایی از آنها، دو کروموزوم فرزند ایجاد می شود [1]. در الگوریتم ژنتیک کلاسیک، از ترکیب یک نقطه ای[۱۴]استفاده می شود. به این صورت که یک نقطه تصادفی انتخاب می شود و هر یک از کروموزوم های والد از محل آن نقطه به دو قسمت تقسیم شده و یکی از قسمت ها را با کروموزوم والد دیگر مبادله می کند. اینکه دو کروموزوم والد با هم ترکیب شوند و در نسل بعد ظاهر گردند یا بدون ترکیب به نسل بعد منتقل شوند توسط عددی تصادفی و مقایسه آن با پارامتری به نام احتمال ترکیب تصمیم گیری می شود.

---

۸ Chromosome

۹ Objective Function

۱۰ Gene

۱۱ Selection

۱۲ Fitness Function

۱۳ Cross over

۱۴ One point

## جهش[۱۵]

بعد از ترکیب عملیات جهش با تغییر تصادفی یک قسمت از هر یک از کروموزم های فرزند انجام می شود [1]. اینکه کروموزوم جهش کند و در نسل بعد ظاهر گردد یا بدون جهش به نسل بعد منتقل شود توسط عددی تصادفی و مقایسه آن با پارامتری به نام احتمال جهش تصمیم گیری می شود.

## ساختار گزارش

در ادامه این گزارش پس از معرفی تغییراتی که در الگوریتم ژنتیک کلاسیک ایجاد شده تا عملکرد آن در حوزه مسائل یافتن نقاط بهینه (کمینه/بیشینه) توابع هدف بهبود دهد، در بخش سوم خلاصه ای از روش ارزیابی و در بخش چهارم نحوه پیاده سازی این الگوریتم ها ارائه شده است. در بخش پنجم نتایج به دست آمده از اجرای الگوریتم های بهبود یافته تشریح و بررسی شده است.

[۱۵] Mutation

# بخش دوم: الگوریتم های پیشنهادی

## بهبود اول – پذیرش حریصانه ترکیب

به عنوان اولین تغییر در الگوریتم ژنتیک کلاسیک، پس از ترکیب دو کروموزوم والد (با درنظر گرفتن احتمال ترکیب) و تشکیل دو کروموزوم فرزند، در صورتی هر یک از فرزندها به جای والد به نسل جدید منتقل می شود که خطای کروموزوم حاصله کمتر از خطای کروموزوم پدر باشد. در غیر این صورت کروموزوم پدر به نسل جدید منتقل می شود.

## بهبود دوم – پذیرش حریصانه نسل جدید

به عنوان تلاشی برای فرار از نقاط بهینه محلی، الگوریتم کلاسیک به گونه ای تغییر داده شده است که به جای تشکیل یک نسل جدید از یک نسل قدیم، چند نسل جدید کاندید تشکیل شود. هر کدام از این نسل های کاندید که دارای کروموزومی با کمترین خطا باشد، به عنوان نسل جدید پذیرفته می شود.

## بهبود سوم – پذیرش حریصانه جهش

پس از جهش یک کروموزوم (با درنظر گرفتن احتمال جهش) در صورتی کروموزوم جهش یافته به نسل جدید منتقل می شود که خطای آن کمتر از خطای کروموزوم جهش نیافته باشد. در غیر این صورت جهش کروموزوم نادیده گرفته می شود.

## بهبود چهارم – ترکیب بهبودهای قبل

در نهایت همه راهکارهای بهبود قبلی شامل پذیرش حریصانه ترکیب، نسل جدید، و جهش به عنوان یک الگوریتم جدید بر روی توابع ارزیابی اجرا خواهد شد.

# بخش سوم: روش ارزیابی

## توابع ارزیابی[۱۶]

در این گزارش از ۴ تابع ارزیابی زیر به عنوان تابع هدف الگوریتم استفاده شده است:

Sphere Function:

$$f(x_1.x_2.\dots.x_n) = \sum_{i=1}^{n} x_i^2$$

Bent Cigar Function:

$$f(x_1.x_2.\dots.x_n) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$$

Rastrigin's Function:

$$f(x_1.x_2.\dots.x_n) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$$

Ackley's Function:

$$f(x_1.x_2.\dots.x_n) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e$$

در همه این ۴ تابع، هدف پیدا کردن کمینه مطلق است که در نقطه صفر اتفاق می افتد [2]:

$$f(x_1 = 0.x_2 = 0.\dots.x_n = 0) = 0$$

همچنین دامنه $x_i$ ها در هر ۴ تابع بازه (-5.12, 5.12) در نظر گرفته شده است.

## دنباله داده های تصادفی یکسان

به منظور ارزیابی نتیجه بهبودهای پیشنهادی در الگوریتم کلاسیک، همه الگوریتم ها برای توابع مختلف از دنباله اعداد تصادفی

---

[۱۶] Benchmark Function

یکسان شروع می کنند، یعنی دنباله های مذکور در ابتدای شروع هر یک از الگوریتم ها بر روی توابع مختلف با seed مشخصی ریست شده و در نتیجه نسل اولیه راه حل های تصادفی اولین تکرار برای همه الگوریتم ها بر روی هر یک از توابع یکسان خواهد بود.

## ارزیابی میزان کاهش خطا در هر نسل

هر یک از پنج الگوریتم کلاسیک و بهبودیافته، ۱ بار بر روی نسل اولیه تصادفی از راه حل ها با مشخصات زیر اجرا شد و پس از تولید و پذیرش هر نسل، میزان خطای بهترین کروموزوم آن نسل (با فرمول زیر) برای ترسیم نمودار و مقایسه ذخیره گردید:

$$Error = (f(x_1.x_2.\dots.x_n) - FuntionGlobalOptimum)^2$$

| عنوان | مقدار |
|---|---|
| تعداد تکرار | ۱ |
| تعداد نسل | ۴۰ |
| تعداد متغیر (n) | ۳۰ |
| طول باینری هر متغیر | ۸ بیت |
| تعداد کروموزوم در هر نسل | ۱۰ |
| احتمال ترکیب | ۰٫۸ |
| احتمال جهش | ۰٫۱ |
| تعداد نسل کاندید (برای الگوریتم های ۲و ۴) | ۵ |

## ارزیابی میزان کاهش خطای نهایی

هر یک از پنج الگوریتم کلاسیک و بهبودیافته، ۵۱ بار بر روی نسل اولیه تصادفی از راه حل ها با مشخصات زیر اجرا شد و پس از اتمام الگوریتم، میزان خطای جواب نهایی (با فرمول زیر) برای مقایسه ذخیره گردید:

$$Error = (f(x_1.x_2.\dots.x_n) - FuntionGlobalOptimum)^2$$

| عنوان | مقدار |
|---|---|

| | |
|---|---|
| تعداد تکرار | ۵۱ |
| تعداد نسل | ۳۰۰ |
| تعداد متغیر (n) | ۳۰ |
| طول باینری هر متغیر | ۸ بیت |
| تعداد کروموزوم در هر نسل | ۱۰ |
| احتمال ترکیب | ۰٫۸ |
| احتمال جهش | ۰٫۱ |
| تعداد نسل کاندید (برای الگوریتم های ۲و ۴) | ۵ |

# بخش چهارم: نحوه پیاده سازی

## زبان و پارادایم برنامه نویسی

پروژه به زبان پایتون Python و به صورت شیء گرا نوشته شده است به گونه ای که در آینده کلاس الگوریتم ژنتیک قابلیت انتشار به صورت کتابخانه ای مستقل از سایر اجزای پروژه داشته باشد. همچنین توابع ارزیابی مورد استفاده در قالب کلاس هایی مشتق شده از یک واسط مشترک نوشته شده اند تا مستقل از پروژه و قابل استفاده در سایر پروژه ها باشند و همچنین قابلیت توسعه و افزودن توابع ارزیابی دیگر وجود داشته باشد.

## ساختار پروژه

پروژه از ۳ ماجول زیر تشکیل شده است:

۱- ماجول اصلی main.py – که وظیفه ایجاد اشیا و فراخوانی متدها به منظور تولید نتایج را دارد.

۲- ماجول الگوریتم ژنتیک geneticAlgorithmSolver.py – که شامل ساختارهای داده ای و عملیاتی مورد نیاز از جمله الگوریتم کلاسیک ژنتیک و امکانات بهبود آن می باشد.

۳- ماجول توابع ارزیابی benchmarkFunctions.py – که شامل ساختارهای داده ای و عملیاتی توابع ارزیابی می باشد.

## کد گذاری کروموزوم ها

در این پروژه هر کروموزوم از اتصال سری معادل باینری هر یک از متغیرها تشکیل می شود. لذا طول کروموزوم برابر است با تعداد متغیر ضربدر طول باینری در نظر گرفته شده برای متغیرها. لازم به ذکر است این مقادیر به عنوان پارامترهای سازنده کلاس GeneticAlgorithmSolver مشخص می گردد.

## تابع هدف و تابع تناسب

در این پروژه توابع ارزیابی مندرج در بخش قبل به عنوان هدف انتخاب می شوند و مسأله اصلی پیدا کردن نقطه بهینه مطلق (در

اینجا کمینه مطلق) است. بر اساس مقدار این توابع بر حسب متغیرهای کد شده در یک کروموزم، احتمال انتخاب آن کروموزوم در میان یک نسل با شرایط زیر محاسبه می شود و در قالب تابع تناسب از آن استفاده خواهد شد:

۱- مجموع مقدار تابع تناسب برای یک نسل (مجموع احتمال انتخاب کروموزوم های یک نسل) برابر ۱ باشد.

۲- کروموزوم هایی که مقدار تابع هدف آن ها کمتر است احتمال بیشتری برای انتخاب داشته باشند. (در توابعی که هدف، یافتن بیشینه مطلق است این موضوع برعکس می باشد.)

۳- مقدار تابع تناسب هر کروموزوم بیشتر از ۰ باشد.

به این منظور برای هر کروموزوم، مقدار تابع هدف آن از بیشترین مقدار تابع هدف کم و به اضافه ۱ می شود؛ این عدد باید تقسیم بر مجموع همین عدد به ازای همه کروموزم های آن نسل گردد تا در بازه (0,1) قرار گیرد.

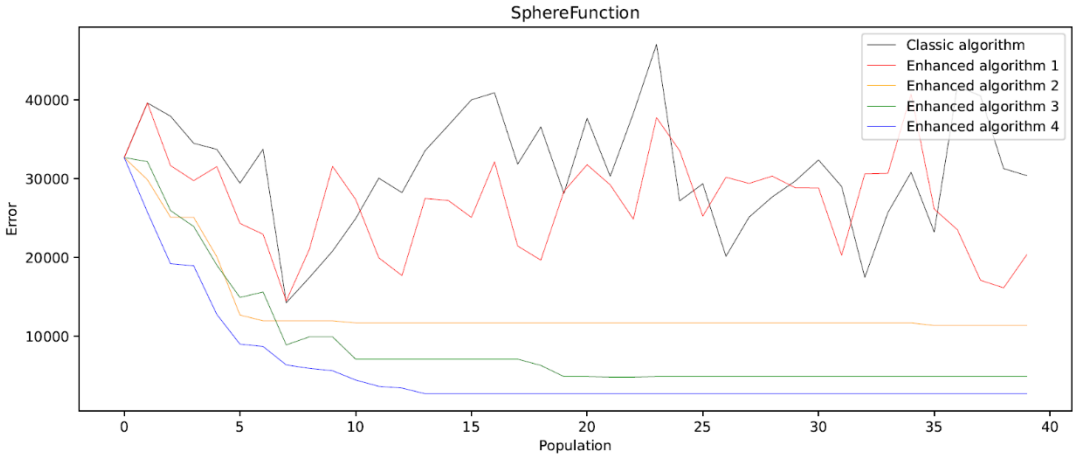
با فرض $f(x_1.x_2.\dots.x_n)$ به عنوان مقدار تابع هدف یک کروموزوم ($x_1.x_2.\dots.x_n$ متغیرهای معدل اجزای کروموزوم هستند)، با داشتن بیشترین مقدار این تابع $Maxf$ و مجموع مقدار این تابع $Sumf$ برای کروموزم های یک نسل و همچنین تعداد کروموزوم های آن نسل $Count$، مقدار تناسب هر یک از کروموزوم ها به صورت زیر محاسبه می شود:

$$fitness(x_1.x_2.\dots.x_n) = \frac{Maxf + 1 - f(x_1.x_2.\dots.x_n)}{Count * (Maxf + 1) - Sumf}$$

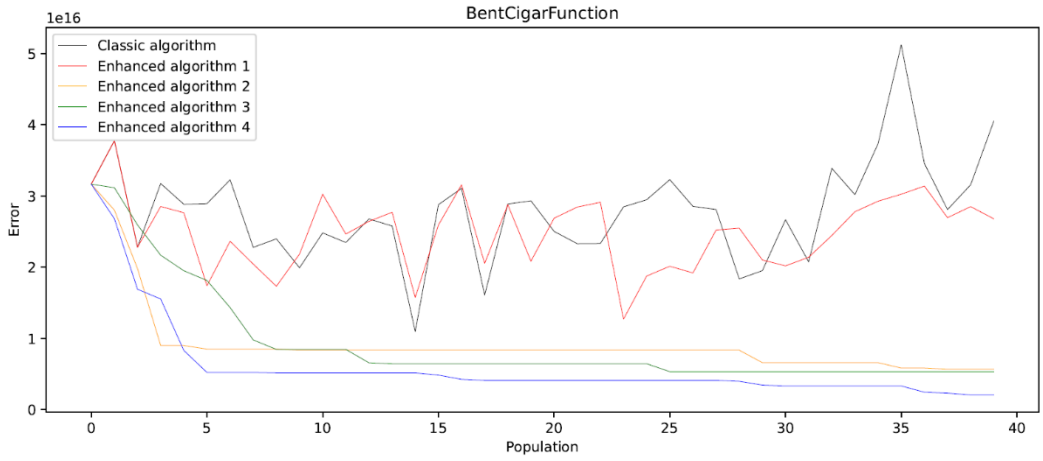
# بخش پنجم: ارائه نتایج

## ارزیابی میزان کاهش خطا در هر نسل

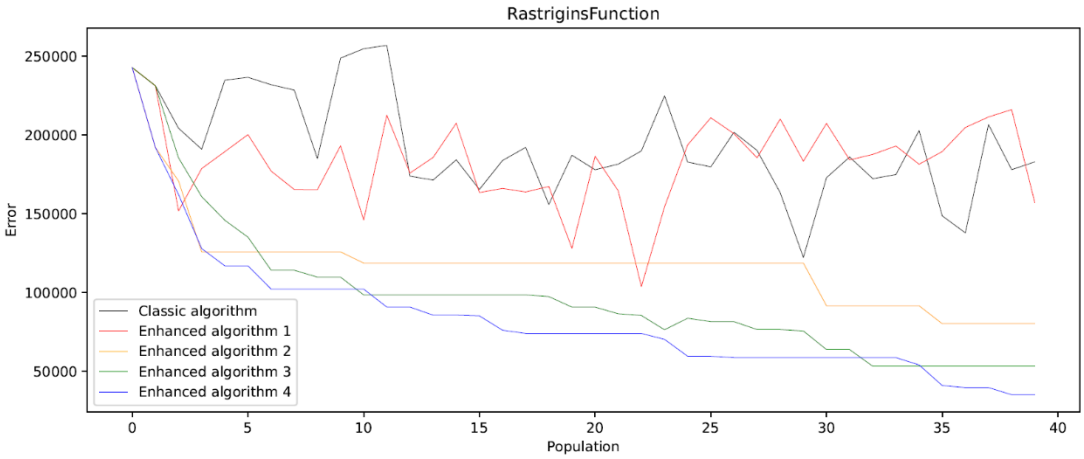
در ادامه خطای بهترین کروموزوم هر نسل در طول اجرای هر یک از الگوریتم ها در یک نمودار برای هر یک از توابع ارزیابی نمایش داده شده است:



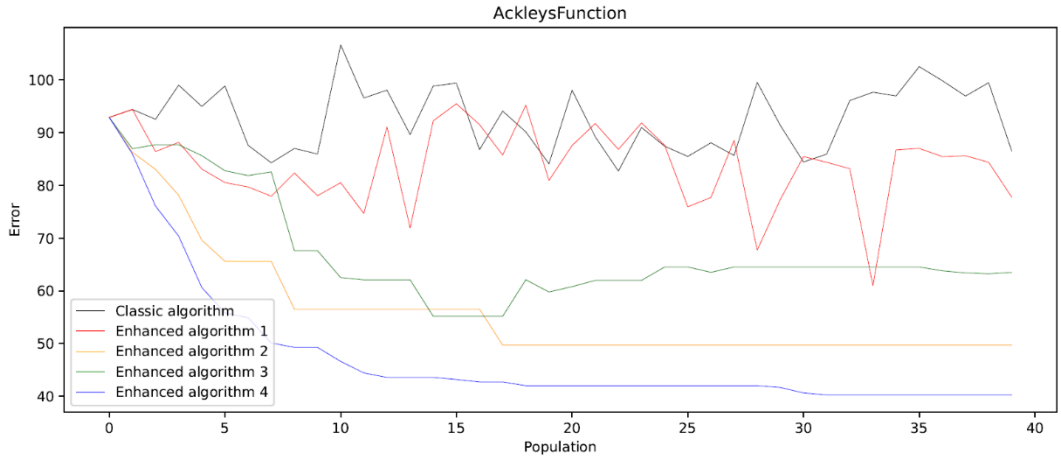

*شکل ۱ نمودار خطای هر نسل در تابع SphereFunction*




*شکل ۲ نمودار خطای هر نسل در تابع BentCigarFunction*




*شکل ۳ نمودار خطای هر نسل در تابع RastriginsFunction*




*شکل ۴ نمودار خطای هر نسل در تابع AckleysFunction*

همان گونه که در نمودارها مشخص است پذیرش حریصانه ترکیب بهبود چندانی در میزان خطای آن نسل نسبت به الگوریتم کلاسیک بوجود نمی آورد. ولی پذیرش حریصانه نسل جدید و پذیرش حریصانه جهش نقش موثری در کاهش خطا در طول نسل ها دارد. مشخص است که ترکیب بهبودهای مذکور هم بهترین نتیجه را در کاهش خطا دارد.

منطقاً پذیرش حریصانه نسل جدید، به دلیل انتخاب نسل جدید از میان چند نسل کاندید، احتمال گیر افتادن الگوریتم در نقاط بهینه محلی را کم می کند، چون بعد از تولید نسل های کاندید از نسل فعلی، نسل های کاندید با هم مقایسه می شوند و نسل بهتر پذیرش می گردد. برتری این روش نسبت به پذیرش حریصانه جهش در توابع Multimodal در بالا مخصوصاً در نمودار AckleyFunction به خوبی مشخص است.

## ارزیابی میزان کاهش خطای نهایی

میانگین خطای نهایی هر یک از الگوریتم ها در جدول زیر ارائه شده است. به منظور مقایسه بهبود ناشی از هر یک از الگوریتم ها، نتایج به تفکیک توابع ارزیابی مرتب شده اند:

| الگوریتم | خطای میانگین |
|---|---|
| کلاسیک | 30418.445 |
| الگوریتم ۱ | 26815.965 |
| الگوریتم ۲ | 2315.917 |
| الگوریتم ۳ | 1777.919 |
| الگوریتم ۴ | 479.244 |

*جدول ۱ خطای میانگین در تابع SphereFunction*

| الگوریتم | خطای میانگین |
|---|---|
| کلاسیک | 2.6937367314007252e+16 |
| الگوریتم ۱ | 2.5756547552519484e+16 |
| الگوریتم ۲ | 1813763555620008.2 |
| الگوریتم ۳ | 1722936617709549.8 |
| الگوریتم ۴ | 561250986225521.9 |

*جدول ۲ خطای میانگین در تابع BentCigarFunction*

| الگوریتم | خطای میانگین |
|---|---|
| کلاسیک | 193471.718 |
| الگوریتم ۱ | 186006.868 |
| الگوریتم ۲ | 61999.591 |
| الگوریتم ۳ | 51393.547 |
| الگوریتم ۴ | 37464.831 |

*جدول ۳ خطای میانگین در تابع RastriginsFunction*

| الگوریتم | خطای میانگین |
|---|---|
| کلاسیک | 91.763 |
| الگوریتم ۱ | 83.086 |
| الگوریتم ۲ | 37.989 |
| الگوریتم ۳ | 38.289 |
| الگوریتم ۴ | 25.750 |

*جدول ۴ خطای میانگین در تابع AckleysFunction*

همانگونه که در بخش قبل اشاره شد پذیرش حریصانه ترکیب بهبود چندانی در میزان خطای الگوریتم نسبت به الگوریتم کلاسیک بوجود نمی آورد، ولی پذیرش حریصانه نسل جدید، پذیرش حریصانه جهش، و الگوریتم ترکیبی نقش موثری در کاهش خطای میانگین تکرارهای متعدد الگوریتم ها بر روی توابع ارزیابی دارد. همچنین مجدداً برتری الگوریتم پذیرش حریصانه نسل جدید نسبت به پذیرش حریصانه جهش علی الخصوص در تابع AckleyFunction به چشم می خورد.

# مراجع

[1] M. M. Lankhorst, "Genetic Algorithms," in *Genetic algorithms in data analysis*, Groningen, 1996, pp. 5-34.

[2] J. J. Liang, B. Y. Qu and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization," 2013.