دانشکده مهندسی

گروه مهندسی کامپیوتر

گرایش هوش مصنوعی

مسئله بهینه سازی تکاملی با محدودیت

استاد:

دکتر مجتبی روحانی

دانشجو:

علی گلی

دی ماه

سال ۱۴۰۲

۱ مسئله

۱.۱ تعریف مسئله

مسئله بیان شده یک مسئله بهینه سازی بنچمارک CEC 2006 است که مسئله هفتم آن را می‌خواهیم حل کنیم. این مسئله شامل ۱۰ متغیر است، ۸ محدودیت دارد و مقادیر تمامی متغیر های ما باید بین بازه (۱۰، ۱۰-) باشند.

۱.۲ حل مسئله

برای حل مسئله دو روش در تمرین گفته شده است. اول اینکه به روش پنالتی تمام محدودیت هارا در نظر بگیریم و دوم اینکه به یک روش ابتکاری مسئله را حل کنیم.

۱.۳ روش ابتکاری

روش ابتکاری من از نوع repair هست. بعد از خروجی گرفتن های متعدد از مسئله با تابع پنالتی متوجه شدم که محدودیت های اول و سوم در اکثر مسائل ارضا نمی‌شوند. ابتدا با دستکاری ژن ها قصد تغییر آن ها را داشتم که باعث خارج شدن الگوریتم ابتکاری از حالت مینیمم تابع هدف شد و سایر محدودیت‌ها نیز ارضا نمی‌شدند. سپس تصمیم گرفتم جریمه ای که برای این دو محدودیت هستند بیشتر کنم تا بتوانم ارضا محدودیت ها را دوباره به الگوریتم بسپارم. یکی از نتایج بسیار جالب:

Best Solution: [1.83279545 3.03617256 7.87859945 4.95448022 1.03316613 1.73419942 1.17850716 9.49945442 7.39149244 9.28981876]

Best Fitness: 39.91717872373613

constraints numbers: [-0.5283871 -6.99713892 -2.21219377 -30.4475971 -5.29474834 -1.56379086 -7.65676997 -47.86670489]

سایر پارامتر ها به شرح ذیل می‌باشند:

Population Size = 100, Mutation Rate = 0.2, generations = 1000

همچنین برای محاسبه تابع fitness مقدار اصلی هدف بعلاوه دو برابر تابع جریمه شده است. خود تابع جریمه نیز همانطور که اشاره شد شامل فاصله تمام محدودیت های ارضا نشده از هدف است؛ ولی محدودیت های ارضا نشده اول و سوم وزن دو برابری دریافت کرده‌اند.

۱.۴ روش توابع جریمه ساده

در این روش برای محاسبه تابع fitness مقدارر اصلی هدف بعلاوه یک وزنی از تابع جریمه می‌شود. در این روش می‌توان مقدار نسل را نیز به تابع fitness افزود تا بعد از چند نسل، تنها به دنبال جواب‌های حول جواب های بهینه باشیم. یکی از جواب های این این نوع روش:

Best Solution: [1.98823577 3.90090396 7.85444889 5.70020639 1.0200157 1.66708815 0.44629052 9.8384603 9.01222268 8.28421763]

Best Fitness: 28.921551479470093

constraints numbers: [9.664734 0.76510771 8.38859989 -33.26978297 3.01120701 -0.05418941 -10.45549743 -28.25366999]

همانطور که مشاهده می‌شود ۳ محدودیت اول نتوانسته‌اند ارضا شوند.

۱.۵ مقایسه نتایج با مقاله

در مقاله ذکر شده در جدول ۳ به ۵ الگوریتم اشاره شده است که هر کدام مسائل مربوط به بنچمارک CEC 2006 را حل نموده‌اند. طبق این مقاله ۲۰۰ نسل برای هر الگوریتم استفاده شده است. اما راجع به سایر مقادیر صحبتی نشده است. الگوریتم من در ۲۰۰ نسل خروجی زیر را تولید می‌کند:

Best Solution: [1.68440049 3.12607943 8.11362255 5.18351708 0.5008416 0.75002384 0.56645697 9.95692709 8.30020491 7.19437334]

Best Fitness: 29.666595204488896

constraints numbers: [5.28097203 2.11945509 7.88923283 -24.26048464 -6.70497335 -2.64618537 -8.52662404 -35.57586235]

سایر پارامتر ها به شرح ذیل می‌باشد:

Population Size = 100, Mutation Rate = 0.2, Penalty Weight = 1

همانطور که مشاهده می‌شود هنوز در ۳ محدودیت اول محدودیت ها نتوانسته اند ارضا شوند. اگر با همین شرایط تابع ابتکاری را فعال نماییم خروجی زیر را مشاهده می‌کنیم:

Best Solution: [2.63495696 2.14374347 7.02988658 6.04525997 0.66278018 1.65126282 1.29983992 9.45979053 8.97145067 8.69686557]

Best Fitness: 47.4065684259718

constraints numbers: [-2.50285983 6.02192419 -1.32864651 -59.33599706 0.83508501 -4.94167536 -9.05021006 -44.59587223]

همانطور که مشاهده می‌شود با اینکه محدودیت های بیشتری ارضا شدند اما مقدار بهترین fitness ماهم افزایش پیدا کرد. در اینجا باید تمامی هایپر پارامتر ها مورد بررسی قرار بگیرند.

۱.۶ هایپر پارامتر ها

در این الگوریتم من هایپر پارامتر های زیادی طراحی شده است که می‌توانند نقش های مختلفی ایفا کنند. در جدول زیر نام و نقش آنها ذکر شده است.

Population size	تعداد افراد جمعیت که ۱۰۰ در نظر گرفته شده است
Mutation rate	میزان جهش که برابر ۰.۲ در نظر گرفته شده است.
Generation	تعداد نسل
Heuristic	استفاده از تابع ابتکاری (به صورت True یا False)
Objective weight	میزان وزن تابع هدف در fitness function
Penalty weight	میزان وزن تابع پنالتی در fitness function
Generation weight	میزان وزن نسل فعلی در fitness function
Heuristic coefficient	میزان وزن خطای محدودیت اول و سوم در صورت فعال بودن تابع ابتکاری

اگر بتوان این مجموعه از هایپر پارامتر ها را به نحوی تعیین کرد که تابع هدف مینیمم شود می‌توان به جواب بهینه نزدیک تر شد.

۲ کد مسئله

۲.۱ کلاس G07

این کلاس شبیه ساز مسئله g07 بنچمارک CEC 2006 می باشد. در آن تابع fitness تعریف شده است و تابع constraint نیز تعریف گشته است. تابع پنالتی در صورت عدم استفاده از قسمت ابتکاری، هر محدودیتی که از صفر بزرگتر باشد به ازای مقداری که از ۰ بزرگتر است را جمع کرده و به عنوان جریمه بر می‌گرداند. در صورت استفاده از قسمت ابتکاری، اگر محدودیت های اول و سوم از ۰ بزرگتر باشند در یک ضریبی آن ها را ضرب می‌کند تا مقدار اررور بیشتری تولید شود و این دو محدودیت که مشکل ساز ترین محدودیت ها هستند را ارضا کند.

۲.۲ کلاس Evolutionary Algorithm

این کلاس یک کلاس مربوط به الگوریتم تکاملی است. یک الگوریتم تکاملی ساده در آن طراحی شده است که شامل تمامی قسمت های یک الگوریتم تکاملی ساده می‌باشد. همچنین این کلاس تابع

fitness را به صورت دیفالت در خود ندارد و ما باید تابع را خارج کلاس تعریف کرده و به عنوان آرگومان به کلاس بدهیم.

۲.۳ تابع custom fitness function

این تابع جمع تابع هدف که باید مینیمم شود، تابع پنالتی و نسل است. هر ۳ این مقادیر می‌توانند ضریبی اختیار کنند که نشان دهنده اهمیت آنهاست. در نتیجه جمع وزن دار آنها به عنوان نتیجه برگشت داده می‌شود.

کد الگوریتم در پیوست خدمت حضورتان ارسال می‌گردد.