به نام خدا

درس : هوش مصنوعی **مدرس : دکتر فدایی**

پروژه اول

علی دارابی – شماره دانشجویی : ۸۱۰۱۰۰۲۶۴

عنوان پروژه و هدف از اجرای آن:

عنوان این پروژه حل مسئله curve fitting به کمک الگوریتم ژنتیک است . هدف ما از این پروژه، یافتن جواب بهینه (بهترین چندجمله ای) برای fit کردن روی نقاط ورودی داده شده به ما است.

بخش ششم ، ارزیابی نتایج :

سوال ۱ :

اگر جمعیت اولیه ما بسیار کم باشد میتواند سبب کاهش تنوع در جمعیت (کروموزوم ها) ما شود در نتیجه رسیدن به جواب بهینه ، زمان خیلی بیشتری لازم دارد یا به طور کل ممکن است که الگوریتم ما نتواند جواب بهینه پیدا کند زیرا که دقت ما کاهش یافته است. اگر جمعیت اولیه ما بسیار زیاد باشد ، برای پیدا کردن جواب زمان بسیار زیادی مصرف می شود و قطعا قدرت محاسباتی خیلی بیشتری نیاز دارد در نتیجه ممکن هست برای اجرای الگوریتم خود نیاز به سیستم هایی با قدرت محاسباتی بالا نیاز داشته باشیم ؛ به همین دلیل باید در انتخاب میزان جمعیت اولیه دقت کنیم و مقداری معقول در نظر بگیریم.

سوال ۲ :

همانطور که در سوال قبل هم گفته شد، افزایش جمعیت قطعا به دقت الگوریتم ما کمک میکند و سبب افزایش دقت الگوریتم میشود ولی از طرفی قدرت محاسباتی لازم برای اجرای الگوریتم هم افزایش میابد و قطعا زمان بیشتری هم مصرف میکند به همین دلیل می توان گفت که این کار سرعت الگوریتم ما را کاهش می دهد.

سوال ۳ :

عملیات crossover در هر بار اجرا ، با ترکیب کردن کروموزوم های نسل قدیم و ایجاد کروموزوم های نسل جدید تاثیر زیادی در الگوریتم ما دارد زیرا که در هر مرحله با ترکیب کروموزوم های شایسته نسل قدیم و ساخت کروموزوم های نسل جدید میتواند باعث پیشرفت کروموزوم ها یا همان جمعیت ما شود و میتواند تعداد کروموزوم هایی که fitness کمتری دارند (کروموزوم های ناشایسته) را کاهش دهد.

عملیات mutation در هر بار اجرا ، تلاش میکند با که با تغییر مقدار ژن ها در یک کروموزوم آن را به یک کروموزوم شایسته تر تبدیل کند ، این عمل را به صورت شانسی انجام میدهد و در واقع ما امیدواریم که با اینکار کروموزوم شایسته تری حاصل شود. علاوه بر این ، mutation به حفظ تنوع جمعیت ما کمک میکند و از یکدست یا همگن شدن سریع جمعیت جلوگیری میکند (اگر جمعیت ما یکدست شود ممکن است نتوانیم جواب بهینه را پیدا کنیم.) ، همچنین ممکن است به دنبال ویژگی خاص در جواب بهینه باشیم ولی کروموزوم های اولیه ما فاقد آن ویژگی باشند ، در نتیجه باید از mutation برای اصلاح جمعیت اولیه خود کمک بگیریم و ویژگی موردنظر را با یک توزیع مناسب در کروموزوم ها اضافه کنیم.

اگر تنها از crossover استفاده کنیم ، ممکن است با مشکلات متفاوتی روبه رو شویم برای مثال ممکن است جمعیت ما بعد از چندین نسل یکدست و همگن شوند و نتوان به جواب بهینه دست یافت یا ممکن است همانطور که گفته شد ویژگی در جواب بهینه بخواهیم که در جمعیت اولیه ما نیست و بدون mutation نمیتوان آنها را اصلاح کرد.

اگر تنها از mutation استفاده کنیم ، هم ممکن است با مشکلات متفاوتی روبه رو شویم ، برای مثال ممکن است کروموزوم های اولیه ما در هر نسل شایسته تر نشوند و یا حتی ممکن است جمعیت ما به سمت جواب بهینه همگرا نشود و نتوانیم به جواب بهینه دست یابیم ؛ همچنین ممکن است کروموزوم های شایسته را با mutation های شانسی و بی هدف از دست بدهیم و الگوریتم ما همانند یک جست و جوی تصادفی رفتار میکند.

سوال ۴ :

مهم ترین عامل در افزایش سرعت رسیدن به جواب ، انتخاب مقادیر معقول برای پارامتر های اصلی مسئله مانند : تعداد جمعیت اولیه ، احتمال crossover ، شیوه انجام crossover مانند: 1-point ، n-point و uniform ، احتمال mutation ، تعداد کروموزوم های شایسته ای که بدون تغییر از یک نسل به نسل بعدی carry میشوند . همچنین عامل دیگری که تاثیر زیادی در سرعت ما دارد ، بهینه سازی و افزایش سرعت توابع و الگوریتم هایی است که ما در حل قسمت های مختلف مسئله استفاده میکنیم.

سوال ۵ :

گاهی اوقات کروموزوم ها بعد از چندین مرحله ، بدون تغییر میمانند ، علت آن این است که میزان تنوع در جمعیت ما کاهش پیدا کرده است و درنتیجه کروموزوم ها مشابه هم و یا یکدست شده اند ؛ به همین سبب الگوریتم ما نمیتواند فضای جست و جوی خود را افزایش دهد و ممکن است به یک جواب غیرمطلوب همگرا شود و جواب بهینه را پیدا نکند.

برای رفع این مشکل باید از mutation کمک بگیریم، هنگامی که جمعیت ما به سمت جواب غیرمطلوب همگرا شده باید به صورت پویا میزان نرخ mutation را طوری تنظیم کنیم که تنوع کروموزوم های ما افزایش یابد ؛ زیرا که mutation به صورت شانسی است و قطعا میزان تنوع مارا تغییر میدهد ، سپس میتوانیم دوباره الگوریتم را پیاده کنیم تا به جواب بهینه دست یابیم.

سوال ۶ :

میتوانیم از چند راه حل ساده استفاده کنیم :

۱ - محدودیت روی تعداد باری که الگوریتم اجرا میشود یا تعدادی مراحلی که ما سپری میکنیم و نسل ها عوض میشوند.

۲ - قرار دادن محدودیت زمانی روی مسئله ، برای مثال اگر در یک دقیقه جواب بهینه پیدا نشد برنامه را متوقف کنیم.

۳ - بهترین کروموزوم هر نسل را در نظر بگیریم و اگر بعد از چندین نسل بهترین کروموزوم نسل آینده نسبت به نسل گذشته به اندازه معقولی بهبود نیافته بود برنامه را متوقف کنیم.

سوال ۷ :

رابطه بین زمان لازم برای پیدا کردن ضرایب و درجه چندجمله ای ، بستگی به این دارد که ما چگونه الگوریتم ژنتیک را پیاده سازی میکنیم ، ولی میتوان همواره گفت که با افزایش درجه چندجمله ای ، زمان پیدا کردن ضرایب قطعا افزایش خواهد یافت ؛ دلیل این اتفاق این است که با افزایش درجه چندجمله ای تعداد ضرایبی که الگوریتم باید پیدا کند افزایش خواهد یافت در

نتیجه فضای جست و جوی ما گسترش میابد و این موضوع قطعا روی نرخ همگرایی الگوریتم تاثیر میگذارد و زمانی که صرف پیدا کردن ضرایب میشود افزایش خواهد یافت .

سوال ۸ :

تعداد نقاطی که به عنوان ورودی به مسئله داده میشود روی عملکرد الگوریتم تاثیر به سزایی دارد ، افزایش تعداد نقاط سبب به منظور اطلاعات بیشتر راجب منحنی مورد نظر است ، در نتیجه ما میتوانیم با دقت بیشتری ضرایب چندجمله ای را پیدا کنیم و یا به اصطلاح fit بهتری پیدا کنیم ؛ اگر تعداد نقاط کاهش یابد قطعا دقت ما در پیدا کردن ضرایب کاهش میابد و ضرایب بدست آمده شاید fit خوبی برای آن نقاط نباشند زیرا که اطلاعات ما از چندجمله محدودتر است.

زمانی که نقاط بیشتری داریم هزینه محاسباتی الگوریتم هم افزایش میابد و قطعا زمان مصرفی تابع fitness ما هم افزایش میابد زیرا که باید هر کروموزوم را با تعداد نقاط بیشتری را بررسی کند. اگر تعداد نقاط کاهش یابد عکس این موضوع اتفاق میافتد.

نرخ همگرایی الگوریتم با تعداد نقاط ورودی تغییر میکند و اگر تعداد نقاط ما کاهش یابد الگوریتم ما زودتر همگرا میشود ولی به احتمال زیاد جواب بدست آمده ، جواب بهینه ای برای مسئله ما نیست. برخلاف اگر تعداد نقاط افزایش یابد الگوریتم ما دیرتر همگرا میشود ولی جواب ما به مراتب شایسته تر از حالت قبل است.