پروژه اول درس رایانش تکاملی

استفاده از الگوریتم ژنتیک در مسئله کوله پشتی

حسین سیم چی

۹۸۴۴۳۱۱۹

استاد: آقای دکتر حامد ملک

۱۳۹۹/۰۸/۲۵

hsimchi74@gmail.com

مقدمه

در مسئله ی کوله پشتی می دانیم که یک کوله داریم و این کوله بر اساس ظرفیتی که دارد بیشتر از وزن محدودی را نمی تواند بپذیرد. سپس **هدف** این است که با برداشتن تعدادی از کالاها که هرکدام از آن ها وزن خاصی را دارند علاوه بر رعایت کردن ورن کوله، کالاهایی را انتخاب کنیم که **بیشترین ارزش ممکن** را داشته باشند تا **درنهایت کوله ای داشته باشیم که از باارزش ترین کالاها تشکیل شده است**. حال می خواهم با استفاده از مفاهیمی که در دروس ساختمان داده و طراحی الگوریتم می دانیم حالت های متفاوتی را برای حل مسئله درنظر بگیریم و این یعنی **نحوه ی انتخاب کالاهای موجود** در بدست آوردن ارزش نهایی تاثیر بسیار زیادی دارد

۱. **بهترین حالت:** در این حالت سعی می کنیم که در **هربار با ارزش ترین کالای** موجود را انتخاب کنیم طوری که محدودیت وزن کوله نقض نشود. طبیعتا در این حالت مطلوب ترین حالت ممکن برای انتخاب کالا رخ داده است و این در صورتی است که یک **جستجوی اولیه** بین کالاها در ابتدا صورت پذیرد تا بدانیم که ارزش هرکالا چیست و سپس هربار با ارزش ترین کالا انتخاب شوند.
۲. **حالت متوسط:** در این حالت فردی که کالاها را انتخاب می کند **بدون جستجوی اولیه** و ندانستن این موضوع که کدام کالا از بقیه با ارزش تر است به صورت **رندوم** یکی از کالاها را انتخاب می کند، فقط در هربار این موضوع را بررسی می کند که شرط وزن کوله حفظ شده باشد. حال سوال این است که اگر به صورت رندوم کالاها را اتخاب کنیم به طور میانگین جمع ارزش کالاهای موجود در کوله چقدر می شود؟
۳. **بدترین حالت:** در این حالت مانند حالت اول فرد در ابتدا جستجو انجام می دهد و هربارکم ارزش ترین کالا را انتخاب می کند تا زمانی که به اندازه ی وزن کوله کالا داشته باشیم. در این حالت که مطلوب مسئله ما نمی باشد نیز می خواهیم بدانیم که در نهایت کوله ی ما چه ارزشی را دارد.

هر سه حالت گفته شده را در این مسئله درنظر گرفته و در انتها نشان می دهیم که اگر هر یک از حالت های فوق رخ دهد به چه مقدار تابع برازنگی خواهیم رسید.

این مسئله را می خواهیم با استفاده از الگوریتم ژنتیک حل کنیم. در نتیجه در ابتدا مفاهیم و مراحل حل مسئله را بیان میکنیم.

حل مسئله با استفاده از الگوریتم ژنتیک

۱. ***تعیین اندازه جمعیت***[1] که در این مسئله به طور دلخواه تعداد جمعیت را برابر ۳۳ درنظر گرفته ایم. که بدین معنا است در هر مرحله ۳۳ کروموزوم یا ۳۳ حالت مختلف از انتخاب کالاها را با جایگشت های متفاوت داریم که تمامی این ۳۳ کروموزوم محدودیت وزن کوله را لحاظ می کنند.

۲. ***تعیین تعداد مراحل*** [2] تکرار الگوریتم که به طور دلخواه ۳۵ درنظر گرفته ایم. یعنی ۳۵ بار الگوریتم ژنتیک را انجام می دهیم تا در مرحله ی آخر بالاترین ارزش را به ازای حالت های بیان شده بدست بیاوریم.

۳. ***مشـــخص نمودن تعداد کروموزوم های انتخابی*** [3] در هر مرحله که هربار مقدار ارزش تک تک کروموزوم ها را بدست می آوریم و ۴ تا کروموزمی که بالاترین ارزش را در هر مرحله دارند؛ انتخاب می کنیم

۴. ***تعیین محدودیت وزن برای کوله***[4] که در این مسئله به طور دلخواه وزن کوله را برابر ۳۰۰ درنظر گرفته ایم.

۵. برای تولید جمعیت اولیه لازم اســت لیســتی را تعریف کنیم که در این لیســت به طول هر کروموزوم مقادیری وجود دارند بین مقدار ۰ و ۱. و دلیل آن نیز بخاطر این است که برای تولید جمعیت **اولیه** در ادامه به مقادیر این لیست نگاه می کنیم و درصورتی که مقدار هر عضو از این لیست کمتر از ۰.۵ باشد مقدار ژن متناظر را برابر ۰ و برای مقادیر بزرگتر از ۰.۵ مقدار ژن را برابر ۱ قرار می دهیم. به عنوان مثال لیست زیر را درنظر بگیرید:

۰.۳	۰.۷۴	۰.۱	۰.۶	۰.۴۷

→

۰	۱	۰	۱	۰

- *نکته :* طول لیســت فوق برابر با طول هر کروموزم می باشــد که در این مســئله به طور دلخواه مقدار ۲۰ را درنظر گرفته ایم

[1] Population Size
[2] Iteration
[3] K
[4] Bag size

۶. با استفاده از کتابخانه ی Random و استفاده از آن جهت تولید اعداد تصادفی اینتیجر، دو لیست به طول ۲۰ ایجاد می کنیم که هر کدام از مولفه ی آن ها نشان دهنده ی یک کالا است و **مقادیر یکی از لیست ها نشان دهنده ی وزن هر قطعه و لیستی دیگر نشان دهنده ی ارزش هر قطعه متناظر است. به عنوان مثال برای فهم بیشتر داریم** :

Weight list	۷۴	۳۶	۴۲	۴۵
Value list	۱۲۱	۱۲	۲۰	۶

به عنوان مثال در مثال بالا، مولفه اول دارای وزن ۴۵ کیلوگرم و ارزشی معادل با ۶ است. اعداد تولید شده با استفاده از مرحله ۵ تولید شده اند.

***** توجه :*** در بیان تحلیل الگوریتم انجام شده، از لیست هایی با طول های غیرواقعی نسبت به کد نوشته شده استفاده شده و تنها هدف از بیان این مثال ها تشریح بهتر و دقیق تر روند انجام کار می باشد. ولی همانطور که در فایل کد موجود قابل برداشت است، طول لیست های ایجاد شده با طول هر عضو از جمعیت یکسان بوده و برابر ۲۰ درنظر گرفته شده است.

۷. **مسئله کوله پوشتی را به صورت مسئله صفر و یک درنظر می گیریم** بدین معنا که هر ژن در یک کروموزوم بیانگر یک قطعه از کالاهایی است که می خواهیم انتخاب کنیم. چنانچه مقدار ژن برابر یک باشد بدین معنا است که اون قطعه انتخاب شده و اگر مقدار خانه یا ژنی برابر صفر باشد بدین معنا است که قطعه مورد نظر انتخاب نشده است. در گام بعدی بجای مشخص کردن دستی اعضای جمعیت اولیه با استفاده از لیست هایی که در مرحله ۶ تولید کردیم و همچنین استفاده از مفاهیم برنامه نویسی، تابعی را به اسم initialise تعریف می کنیم تا فرایند بیان شده در گام ۵ را به صورت اتوماتیک انجام دهد. درنتیجه از این تابع تنها برای جمعیت اولیه استفاده می شود و بدون گونه است که بر اساس اندازه جمعیتی که در گام اول مشخص کردیم شروع به تولید جمعیت اولیه می کنیم. **توجه داشته‌باشید‌به دلیل اینکه جمعیت اولیه را در گام اول برابر ۳۳ درنظر گرفته ایم درنتیجه ۳۳ لیست یا به اصطلاح کروموزوم با طول ۲۰ را خواهیم داشت. که هرکدام از این لیست ها دارای مقادیر صفر و یک است.**

۸. تابعی به اسم Evaluate تعریف می کنیم که برای هر لیست بررسی می کند که جمع مقادیر وزن های کالاهای انتخاب شده چقدر می باشد. در صورتی که جمع وزن های انتخاب شده بیشتر از وزن قابل تحمل کوله باشد، ارزش کروموزوم انتخاب شده را برابر صفر قرار می دهد و در صورتی که وزن قابل تحمل کوله رعایت شده باشد مجموع ارزش کالاهای کروموزوم را نیز بدست می اوریم و در لیستی به اسم لیست Fitness ذخیره می کنیم. در نتیجه **خروجی تابع Evaluate لیستی به طول اندازه جمعیت می باشد که هر مولفه ی آن بیانگر صفر یا مقداری خواهد بود که مقدار صفر بیانگر این موضوع است که کروموزم مدنظر وزن کوله را رعایت نکرده است درنتیجه خروجی تابع برازنگی برای این کروموزوم صفر شده است و در صورتی که مقداری داشته باشیم، مقدار بیانگر ارزش کروموزوم مدنظر بوده است که این ارزش بیانگر مجموع ارزش کالاهای انتخاب شده در کروموزوم انتخابی است.** به عنوان مثال فرض کنید خروجی تابع Evaluate لیستی مانند زیر است که این لیست را برابر لیست برازندگی یا فیتنس می دانیم

Fitness List	۰	۱۴۵	۰	۰	۰	۶۹۸	۰	۴۵	۱۲	۰

به عنوان مثال لیست فیتنس بالارا درنظر بگیرید؛ در این لیست قابل مشاهده است که برخی از خانه ها مقدار صفر و برخی دیگر مقدار مخالف صفر دارند. هر خانه از این لیست نشان دهنده ی ارزش یک کروموزم از جمعیت اولیه است. *خانه هایی که مقدار مخالف صفر دارند، خانه هایی هستند که وزن کروموزوم آن از وزن کوله پشتی بیشتر نشده درنتیجه ارزش آن را بدست آورده و در لیست قرار می دهیم ولی برای کروموزوم هایی که وزن آن ها از وزن کوله بیشتر است مقدار صفر را برمی گردانیم.*

*** توجه : در این مسئله تابع فیتنس یا برازنگی را برابر جمع ارزش کالاهای انتخاب شده تعریف می کنیم به شرط اینکه شرط وزن کوله حفظ شود.

۹. در مرحله بعد، تابعی به اسم **انتخاب والدین**[5] انتخاب می کنیم که وظیفه ی این تابع دریافت لیست برازندگی و انتخاب ۴ تا از کروموزوم یا والدینی است که بیشترین مقدار را در لیست فیتنس داشته اند.

[5] Parent Selection

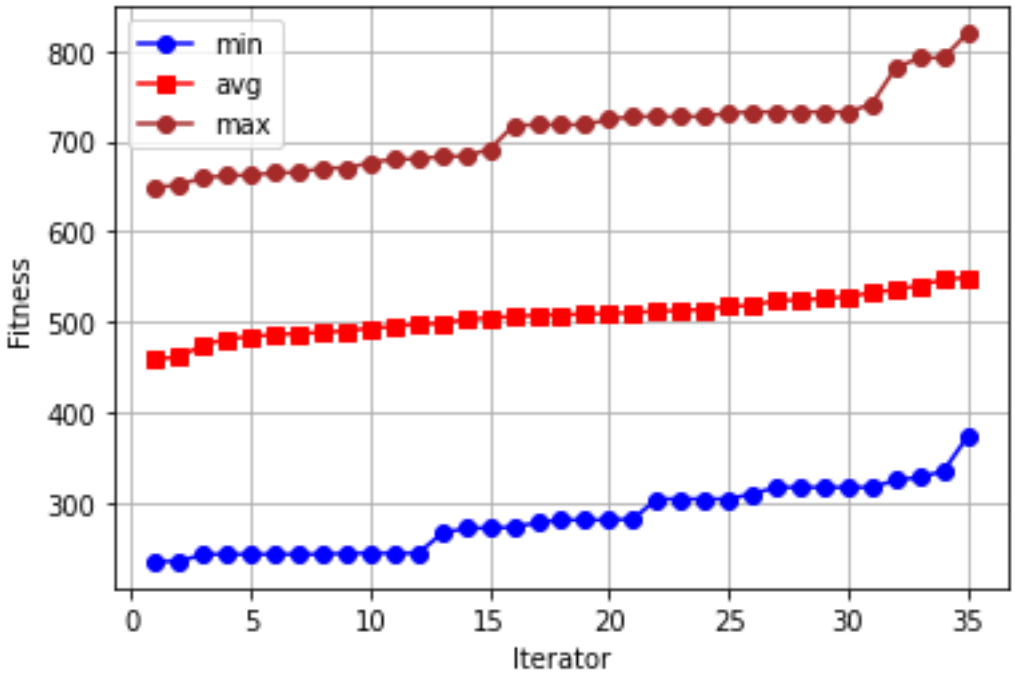
۱۰. تابعی با عنوان recombination تعریف می کنیم که دو به دو کروموزوم هایی که از مرحله ۹ انتخاب شده است را دریافت کرده و رندوم از مکانی خاص شروع به انجام عمل ترکیب کرده و فرزندانی جدید را تولید می کنیم

۱۱. سپس تابعی به اسم Mutation را تعریف می کنیم که به طور شانسی عمل میوتیشن را بر روی یکی از فرزندان و یکی از ژن های آن انجام می دهد بدین صورت که اگر مقداری برابر صفر باشد آن را برابر یک و بالعکس تغییر می دهد.

۱۲. در گام آخر نیاز داریم با استفاده از یک حلقه While، به تعداد مراحلی که در گام ۲ مشخص کردیم مراحل ۸ تا ۱۱ را ادامه دهیم.

نتایج و بحث

در این بخش به بررسی نتایج بدست آمده می پردازیم و خواهیم دید که خروجی هر بخش از مراحل گفته شده به چه صورتی خواهد بود



شکل ۱) خروجی الگوریتم به ازای ۳ حالت بیان شده

همانطور که از شکل ۱ قابل برداشت است، مسئله را برای ۳ حالت بیان شده در بخش مقدمه حل نموده ایم و در مراحل مختلف در هر سه حالت مقدار تابع برازنگی ما افزایش یافته است ولی بسته به اینکه برای انتخاب کالا از کدام روش استفاده کنیم نتیجه ما متفاوت بوده است.

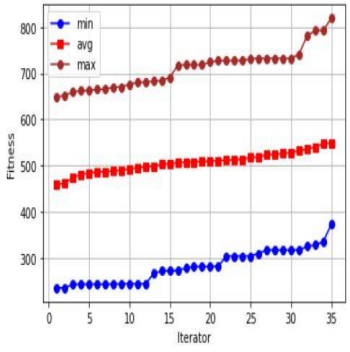
در شکل۱، زمانی که فردی بخواهد کالاها را انتخاب کند اگر در ابتدا جستجو انجام دهد و هربار با ارزش ترین کالا را انتخاب کند و البته هربار چک کند که محدودیت وزن کوله نقض نشده باشد نمودار مربوط به رنگ قهوه ای بدست می آید که در انتها به ماکزیمم ارزش میرسیم. که این حالت مطلوب ترین حالت مسئله خواهد بود و ماهم برای حل این مسئله به دنبال همچین رویکردی هستیم. ولی بدترین حالت و حالت متوسط را هم درنظر گرفته ایم تا مقایسه ای بین این حالت ها داشته باشیم.

در حالت متوسط اگر بدون جستجو، هربار به صورت رندوم یکی از کالاها را انتخاب کنیم نمودار قرمز رنگ بدست می آید که در انتهای مرحله ۳۵، به مقدار ارزش بین ۵۰۰ تا ۶۰۰ میرسیم که متوسط بهترین حالت و بدترین حالت را نشان می دهد یعنی اگر هربار رندوم یکی از کالا ها را انتخاب کنیم و تنها شرط وزن رو رعایت کنیم به همچین ارزشی بعد از ۳۵ مرحله خواهیم رسید.

در بدترین حالت هم هربار کالا با کمترین ارزش را انتخاب می کنیم و شرط وزن را چک می کنیم که در این حالت هم در بدترین حالت ممکن بعد از ۳۵ مرحله نهایتا به کوله ای با ارزش حدود ۴۰۰ خواهیم رسید.

در شکل ۲ نیز خروجی کد قابل مشاهده است:

```
Random Weight List is : [35, 2, 27, 10, 19, 14, 3, 8, 21, 19, 18, 8, 19, 5, 34, 4, 25, 7, 2, 33]
Random value List is : [5, 76, 15, 76, 27, 69, 53, 7, 77, 89, 66, 69, 57, 38, 78, 34, 5, 68, 8, 82]
The First generation of population list is  : ['0010110001001011111', '0001010010100010101001', '1010001110000010011', '1001111011011001001', '0110100111010100010', '1010001111011011
Fitness list for each Chromosome in the population list is : [532, 453, 281, 644, 406, 565, 732, 317, 514, 437, 573, 596, 444, 546, 665, 489, 491, 482, 342, 635, 486, 387, 566, 355, 56
Selected parents with the most values are : ['0010110001001011111', '0001010010100010101001', '1010001110000010011', '1001111011011001001', '0110100111010100010', '1010001111011011111
children is generated with cross over operation : ['0010110001001001101001', '0001010001001011111111', '0001001110000010011', '1010010010100010101001', '1010001110000010001', '1001111011016
random mutation on children is : ['1011000111010100000 1', '1111111000110101011 0', '1001001000110100010 0', '1011100010010011010 0']
```



شکل ۲) خروجی الگوریتم ژنتیک برای مسئله کوله پشتی