مسئله کوله پشتی با GA

امیرحسین انتظاری ۶۱۰۳۹۹۱۰۳

امیرعلی امینی ۶۱۰۳۹۹۱۰۲

**تعریف مسئله:**

مسئله کوله‌پشتی ۰-۱ یک مسئله بهینه‌سازی در علم کامپیوتر است که در آن باید مجموعه‌ای از اشیاء با ارزش و وزن داده شده را درون یک کوله‌پشتی با ظرفیت مشخص قرار داد. هر شیء دارای وزن و ارزشی است و باید تعیین کنیم کدام شیء‌ها را درون کوله‌پشتی قرار دهیم تا مجموعه‌ای با بیشترین ارزش را داشته باشیم، در حالی که وزن کل شیء‌ها ظرفیت کوله‌پشتی را نباید بیشتر کند.

در این مسئله، هر شیء را می‌توان به دو صورت گرفته یا نگرفته در نظر گرفت. به عبارت دیگر، مجموعه شیء‌های قابل انتخاب برای قرار دادن در کوله‌پشتی یا ۰ (شیء را نگرفتن) یا ۱ (شیء را گرفتن) است.

هدف در این مسئله انتخاب مجموعه‌ای است که مجموع ارزش شیء‌های داخل آن بیشینه شود، در حالی که وزن کلی آن مجموعه نباید بیشتر از ظرفیت کوله‌پشتی باشد. این مسئله به عنوان یک مسئله بهینه‌سازی معروف است و الگوریتم‌های مختلفی برای حل آن وجود دارد.

ما در این پروژه سعی داریم با الگوریتم ژنتیک، جواب بهینه ای برای آن پیدا کنیم.

**الگوریتم ژنتیک:**

الگوریتم ژنتیک یک روش محاسباتی مبتنی بر تکامل است که بر اساس فرآیندهای زیست‌شناختی و تکاملی در طبیعت الهام گرفته شده است. این الگوریتم برای حل مسائل بهینه‌سازی و جستجو در فضای مسئله استفاده می‌شود.

الگوریتم ژنتیک بر مبنای مفهوم تکامل طبیعی که توسط چارلز داروین مطرح شده است، ساخته شده است. در طبیعت، تکامل طبیعی فرآیندی است که توسط انتخاب طبیعی، تنوع ژنتیکی و میراث ژنتیکی در جمعیتی از موجودات زنده رخ می‌دهد. الگوریتم ژنتیک نیز این فرآیندها را شبیه‌سازی می‌کند تا به راه‌حل‌های بهینه در فضای مسئله برسد.

الگوریتم ژنتیک از تعدادی از عناصر به عنوان "جمعیت" شروع می‌کند که هر یک از آن‌ها یک "جواب" از مسئله را نشان می‌دهند. هر حواب یک "کروموزوم" است که معمولاً با استفاده از رشته‌ای از صفر و یک‌ها نمایش داده می‌شود. هر کروموزوم پارامتر های خود را بر اساس هدف مسئله با استفاده از تابع برازش ارزیابی می‌کند.

مراحل اصلی الگوریتم ژنتیک عبارت‌اند از:

1. ایجاد جمعیت اولیه تصادفی از کورموزوم ها.

2. ارزیابی برازش هر کورومزوم بر اساس پارامترهای تعریف شده.

3. تولید نسل بعدی با استفاده از عملیات شامل ترکیب ژنتیکی (تهیه‌ی نسل جدید از ترکیب دو یا چند جهش) و ایجاد نسل جدید.

4. انتخاب جهش‌هایی برای تولید نسل بعدی.

5. تکرار مراحل 2 تا 4 تا رسیدن به شرایط خاتمه مسئله یا دستیابی به جواب بهینه.

با تکرار مراحل فوق، الگوریتم ژنتیک به تدریج به راه‌حل‌های بهینه نزدیک‌تر می‌شود و می‌تواند در مسائل بهینه‌سازی پیچیده و فضاهای جستجوی بزرگ عملکرد خوبی داشته باشد. الگوریتم ژنتیک در مسائل مختلفی مانند یافتن کوتاه ترین مسیر، برنامه‌ریزی زمان،بهینه‌سازی تولید و … مورد استفاده قرار می‌گیرد.

**روش حل مسئله:**

برای حل مسئله کوله پشتی ۰-۱ با الگوریتم ژنتیک، ابتدا باید تعیین کنیم که جواب های مسئله ما چگونه باید در قالب کورموزوم پیاده سازی شوند. یک راه پیشنهادی این است که بیایم یک جواب مسئله که شامل تعدادی شی هست را مستقیما به عنوان کورموزوم انتخاب کنیم. کوروموزوم گفته شده به صورت زیر خواهد بود:

Chromosome = [80, 82, 85, 70, 72, 70, 66, 50, 55, 25, 50, 55, 40]

که اعداد درون آرایه، وزن اشیا میباشند. روش دیگر برای نمایش راه حل در کورموزوم، استفاده از نمایش باینری است، که به نظر من روش بهتری است چرا که اگر چند شی با وزن تکراری داشته باشیم، پیاده سازی چک کردن موجود بودن آن در کوله پشتی راحت تر است. کوروموزوم گفته شده به صورت زیر خواهد بود:

Chromosome = [0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0,]

پس از انتخاب نحوه نمایش کوروموزم، ابتدا یک جمعیت اولیه از کورموزوم ها ایجاد میکنیم. من اینکار را با کوروموزم اولیه با مقادیر ۰ انجام دادم. تعداد جمعیت اولیه نیز بسته به بزرگی ورودی مسئله تغییر میکند. سپس، مقدار فیتنس هر کورموزوم را در یک آرایه ذخیره میکنیم که خانه i ام آرایه مربوط به کورموزوم i ام از جمعیت اولیه است.

حال باید یک تابع بازترکیب(crossover) برای ترکیب دو کورموزوم با یکدیگر ارائه دهیم. من از روش بازترکیب دو نقطه ای برای تابع بازترکیب استفاده کردم. در این روش ابتدا بچه اول برابر با والد اول و بچه دوم برابر با بچه دوم میشود، سپس دو اندیس رندوم انتخاب میشود و از اندیس اول تااندیس دوم کوروزموم والد دوم به بچه اول و از اندیس اول تا اندیس دوم والد اول به بچه دوم ارث می رسد. در نهایت بچه با برازش بیشتر انتخاب می شود.

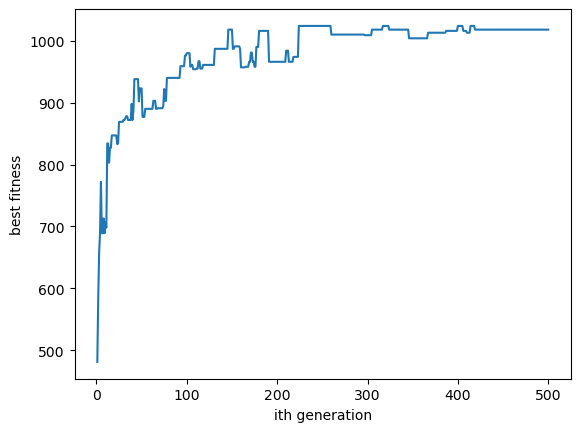
در کنار تابع بازترکیب باید تابع جهش را نیز تعریف کنیم، تا بتوانیم به الگوریتممان پویش بدهیم. از بین روش های جهش موجود، من دو روش جا به جایی و معکوس سازی بیت را انتخاب کرده ام. هر چند که روش جا به جایی برای مسئله با کورموزوم دودویی خیلی مناسب نیست و در الگوریتم نیز استفاده نشده است. در روش معکوس سازی بیت، روی کوروموزوم حرکت کرده، و با احتمال یک پارامتر تصادفی، بیتی که روی آن هستیم معکوس می شود. من با پارامتر ۰.۲ جواب های خوبی گرفتم.

حال که تابع بازترکیب و جهش را تعریف کردیم، میتوانیم به تعریف تابع انتخاب بپردازیم. از بین روش های موجود برای تابع انتخاب، من روش تورنومنت و رنک را پیاده سازی کرده ام. هر چند که روش رنک بسیار انتفاع را زیاد میکند و سریع در نقطه بهینه محلی گیر میوفتیم و در نتیجه استفاده ای از آن نکردم. در روش تورنومنت، ابتدا n عضو برای والد اول، سپس m عضو برای والد دوم انتخاب کردم، و بهترین والد ها را از هر دوی این دسته برای بازترکیب انتخاب میکنم. پس از تولید هر بچه، کوروموزوم را به تابع جهش داده و با احتمال ۰.۲ جهش میدهیم.

نکته ای که در هنگام بازترکیب و جهش همواره توجه میکنیم، قابل قبول بودن(feasible) جواب هاست. به طوری که هر سری، چک میکنیم که آیا بچه تولید شده قابل قبول است یا خیر.

در نهایت عملیات انتخاب را تا زمان برقراری شرط مورد نظر که من برای این الگوریتم، رسیدن به جواب پیشنهادی و محدودیت نسل را انتخاب کرده ام، ادامه میدهیم.

تحلیل نمودار:

نمودار برای تست کیس اول:

همانطور که ملاحظه میکنیم، در زمان خیلی خوبی، الگوریتم جواب اولیه را به جواب خواسته شدن میرساند. همچنین میبینیم که هر چه به جلو تر میرویم، کمتر میتوانیم انتفاع را زیاد کنیم و همچنین پویش، جواب بهتری به ما نمیدهد.

نمودار برای تست کیس دوم:

ملاحظه میکنیم که جواب اولیه که ۱۶۰۰ می باشد، به صورت به جواب خواسته شده نزدیک میشود(طی چند نسل کوتاه) اما از آن پس به آرامی میتواند به بهترین جواب نزدیک شود. و اینکار عموما با پویش انجام میگیرد.