حل مساله زمان بندی ماشین های موازی با استفاده از الگوریتم ژنتیک در متلب

برای حل مساله زمان بندی ماشین های موازی، به عنوان مثال تقسیم وظایفی بین ماشین ها که بیشترین سود را در کمترین زمان داشته باشیم دیتا بیسی در اختیار داریم. برای حل چنین مسئله هایی استفاده از الکوریتم ژنتیک رهیافت مناسبی است چرا که در این مسائل از آنجایی که سود و زمان حل مسئله برای هر ماشین امری کاملاً تصادفی است لذا این سیستم تصادفی که از هیچ فرمولی پیروی نمیکند را نمیتوان فقط با یک فرمول به جواب رساند بلکه باید همه ی حالات مختلف را با یک جهش تصادفی تست کرد ولی این تصادف بر اساس امور تصادفی هدایت شده مانند چرخ رولت در الگوریتم ژنتیک به ما اجازه میدهد جستجویی هدفمند در میان امور تصادفی در جهت ماکزیمم سازی یک یا چند اتفاق را رقم بزنیم.

پارامترهایی که برای حل مساله مورد نیاز است شامل موارد زیر است:

- تعداد ماشین ها
- تعداد فعالیت ها
- زمان پردازش هر فعالیت بر روی هر ماشین
- زمان راه اندازی هر فعالیت نسبت به فعالیت های دیگر بر روی هر ماشین
- سود (که به ما نشان میدهد که برای هر عمل چقدر سود خواهیم داشت)
- زمان مرگ (که در صورتی که ان عملیات انجام نگیرد در زمان مقرر آن عملیات از دست خواهد رفت)

نمایش خروجی به صورت زیر باشد:

```
=========================================================
 Best Solution = 44  48  39  35  24  54  50  15   7  21  32   2  28  38  19   3   1  34  29   8  10  52  46  41  37  31  12
 Best Fitness = 113
 Time = 11.726
=========================================================
 Machine = 1 Time = 178 Benfit = 73 Rejected = 4 Job = 42  13  27  22  33   6  18  45  11  14  47
 Machine = 2 Time = 249 Benfit = 136 Rejected = 0 Job = 50  15   7  21  32   2  28  38  19   3   1  34  29   8  10
 Machine = 3 Time = 213 Benfit = 106 Rejected = 2 Job = 46  41  37  31  12  26   5   9  20  25  23
 Machine = 4 Time = 133 Benfit = 34 Rejected = 1 Job = 44  48  39  35  24
 Machine = 5 Time = 166 Benfit = 55 Rejected = 2 Job = 40  17  16  49  30  36  43   4
=========================================================
```

مراحل الگوریتم ژنتیک:

- تنظیم پارامتر های اولیه
- تولید جمعیت اولیه به صورت تصادفی
- شروع حلقه اصلی
- انجام عملیات تقاطع
- انجام عملیات جهش
- یک کاسه کردن تمام جواب ها (والدین و فرزندان)
- انتخاب بهترین ها به عنوان والد نسل بعد
- در صورت برقرار نبودن شرط توقف به ابتدای حلقه اصلی الگوریتم برگردد
- نمایش نتایج

روش متداول پیاده سازی الگوریتم ژنتیک بدین ترتیب است که:

- مجموعه ای از فرضیه ها که population نامیده میشود تولید وبطور متناوب با فرضیه های جدیدی جایگزین میگردد.
- در هر بار تکرارتمامی فرضیه ها با استفاده از یک تابع تناسب یا Fitness مورد ارزیابی قرار داده میشوند . آنگاه تعدادی از بهترین فرضیه ها با استفاده از یک تابع احتمال انتخاب شده و جمعیت جدید را تشکیل میدهند.
- تعدادی از این فرضیه های انتخاب شده به همان صورت مورد استفاده واقع شده و مابقی با استفاده از اپراتورهای ژنتیکی نظیر Crossover وMutation برای تولید فرزندان بکار میروند.

یک الگوریتم GA دارای پارامترهای زیر است:

GA(Fitness,Fitness_threshold,p,r,m)

- Fitness :تابعی برای ارزیابی یک فرضیه که مقداری عددی به هر فرضیه نسبت میدهد
- Fitness_threshold :مقدار آستانه که شرط پایان را معین میکند
- p :تعداد فرضیه هائی که باید در جمعیت در نظر گرفته شوند
- r:در صدی از جمعیت که در هر مرحله توسط الگوریتم crossover جایگزین میشوند
- m:نرخ mutation

- Initialize :جمعیت را با تعداد p فرضیه بطور تصادفی مقدار دهی اولیه کنید.

- Evaluate :برای هر فرضیه h در p مقدار تابع Fitness(h) را محاسبه نمائید.
- تا زمانیکه $[\max_h \text{Fitness}(h)] < \text{Fitness_threshold}$یک جمعیت جدید ایجاد کنید.
- فرضیه ای که دارای بیشترین مقدار Fitness است را برگردانید.

مراحل ایجاد یک جمعیت جدید بصورت زیر است:

۱. select :تعداد (1-r)pفرضیه از میان P انتخاب و به P_sاضافه کنید .احتمال انتخاب یک فرضیه h_iاز میان Pعبارت است از:

$$P(hi) = \text{Fitness}(hi) / \Sigma j \text{ Fitness}(hj)$$

۲. Crossover :با استفاده از احتمال بدست آمده توسط رابطه فوق، تعداد (rp)/2زوج فرضیه از میان P انتخاب و با استفاده از اپراتور Crossoverدو فرزند از آنان ایجاد کنید .فرزندان را به P_s اضافه کنید.

۳. Mutate :تعداد m درصد از اعضا P_sرا با احتمال یکنواخت انتخاب و یک بیت از هر یک آنها را بصورت تصادفی معکوس کنید

۴. Update: $P \leftarrow P_s$

۵. برای هر فرضیه h در P مقدار تابع Fitness را محاسبه کنید