AI-HW4

فرزان رحمانی 99521271

۱) چون این مسئله دنبال کمترین تعداد رنگ است پس یک مسئله optimization می‌باشد که می‌توان با local search آن را حل کرد. یکی از الگوریتم‌های خوب local search هم genetic می‌باشد پس این مسئله را با آن حل می‌کنیم.

n = تعداد رأس‌ها	$n=15$ ← تعداد رأس‌ها
k = تعداد رنگ‌ها	$k=8=7+1$

7 ← بیشترین درجه درین رأس‌ها

آ) تولید جمعیت اولیه

encoding technique: جمعیت ما از یک سری gene یا chromosome یا فرد تشکیل شده است. هر gene یک آرایه به طول $n=15$ می‌باشد که هر خانهٔ آن یک رنگ دارد. رنگ‌ها (color, label) یک عدد از 1 تا $k=8$ هستند که آرایه را پر می‌کنند. نسل اول ~~جمعیت اولیه adjacency matrix~~ جمعیت ما دارای 6 عضو است که اعداد اعضای آن با اعداد تصادفی از 1 تا $k=8$ پر شده‌اند که هرکدام یک نوع رنگ کردن گراف را نشان می‌دهند اگرچه اشتباه باشد. در ابتدا k را یک واحد بیشتر از بزرگترین درجه درمیان رأس‌ها در نظر می‌گیریم به این علت که یک حد بالا برای جواب مسئله است.

$$\underbrace{\chi(G)}_{\text{chromatic number}} \leq \underbrace{\Delta(G)+1}_{\substack{\text{بیشترین درجه} \\ \text{درین رأس‌ها}}} \Rightarrow \chi(G) \leq 8 \Rightarrow k=8$$

به طور مثال بعد از اجرای توضیحات بالا به این جمعیت می‌رسیم.

first-generation

1. [2,3,8,5,7,6,1,2,3,4,7,6,5,1]
2. [1,2,7,8,4,5,6,1,2,2,1,6,5,5]
3. [7,5,4, ... ,1]
4. [5,2,8 - - - - - - - - - - - - - - ,2]
5. [4,1,3, - - - - - - - - - - - - - - ,8]
6. [3,2,7, - - - - - - - - - - - - - - ,7]

population size = 6

طبق فرض سؤال

الگوریتم برای کاهش k

ما می‌خواهیم k را minimum کنیم. الگوریتم ما با حد بالای $k=8$ شروع می‌شود. اگر به نسلی رسیدیم که مقدار fitness آن صفر باشد یعنی یک رنگ آمیزی قابل قبول برای گراف با شد k را یک واحد کاهش و الگوریتم را دوباره اجرای می‌کنیم تا یک رنگ آمیزی قابل قبول با $k-1$ رنگ پیدا کنیم. این process را اینقدر انجام می‌دهیم تا اینکه نتوانیم یک رنگ آمیزی قابل قبول برای گراف پیدا کنیم. در این مرحله یا خود جواب یا عدد نزدیک به جواب را پیدا می‌کنیم.

1

ب) محاسبه Fitness (evaluation function)

برای تعریف fitness function، برای هر یال که دارای رأس هایی با رنگ یکسان است 1 واحد جریمه در نظر می گیریم.

penalty(i,j) = 1 if there is an edge between i and j and they have same color
= 0 otherwise

بنابراین fitness value هر کروموزوم (یک عضو جمعیت) برابر است با جمع مقدار penalty function (تابع جریمه) برای همه یال های graph.

$$fitness = \sum penalty(i,j)$$

هرچه مقدار این تابع (fitness) کمتر باشد آن عضو بهتر است و شرایط ما را بهتر حفظ می کند و اگر صفر باشد شرط سؤال را نقض نمی کند و جواب قابل قبول است. اگر یک جمعیت را در نظر گیریم براساس آن می توانیم عضوهای قوی تر و بهتر نسل را به ترتیب شناسایی کنیم.

i-th generation
1. [...] $f_1 = 1$
2. [...] $f_2 = 3$
3. [...] $f_3 = 4$
4. [...] $f_4 = 5$
5. [...] $f_5 = 7$
6. [...] $f_6 = 10$

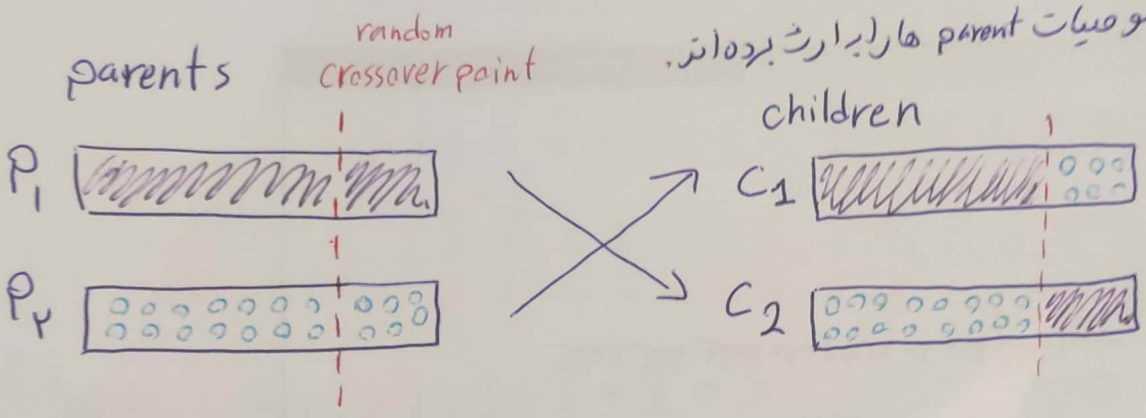
ج) Mutation , Crossover

برای عملیات crossover ابتدا باید عملیات selection را روی نسل قبل انجام دهیم. این عملیات طوری انجام می شود که کروموزوم fit تر احتمال بیشتری برای parent بودن داشته باشد.
دو الگوریتم می توان استفاده کرد:

1. Tournament Selection: ابتدا جمعیت را مخلوط می کنیم به شکل تصادفی. سپس آن ها را در گروه های دو نفره (tournament size = 2) می ریزیم و کروموزوم fit تر (fitness کمتر) انتخاب می شود. به این ترتیب اگر جمعیت N باشد $\frac{N}{2}$ نفر انتخاب شده اند. لذا یک بار دیگر آن را تکرار می کنیم تا سایز جمعیت (N) حفظ شود. این روش تضمین می کند که 2 fittest gene بار آمده است و least fit gene انتخاب نمی شود.

2. Roulette-Wheel Selection: این روش براساس fitness value و احتمال است. به این گونه که roulette wheel به هر کروموزوم یا gene یک تکه اختصاص می دهد که سایز آن تکه متناسب با fitness value آن می باشد. سپس roulette wheel را می چرخانیم تا یک عضو را بدهد. wheel را N بار می چرخانیم و هر جا که ایستاد یک ژن یا کروموزوم را نشان می دهد. به این ترتیب N عضو select می شوند

حال که parent ها را به یکی از دو روش انتخاب کردیم crossover را انجام می دهیم. می خواهیم co parent ها را ترکیب کنیم و child های جدید ایجاد کنیم. یک نقطه (index of array) به طور تصادفی از آرایه parent ها انتخاب می کنیم (crossover point). رنگ های سمت راست این نقطه را جابه جا می کنیم و فرزندان جدید ایجاد می شوند که خصوصیات parent ها را به ارث برده اند.



Mutation یک جهش کوچک برای فرار کردن از اکسترمم محلی است. معمولاً با احتمال کمی انجام می شود. مثلاً می توان با احتمال 10% یا 20% اجازه mutation داد. این عمل یکی از رأس ها (1 تا N) را به طور تصادفی انتخاب می کند و رنگ آن را تغییر می دهد. این کار علاوه بر فرار از local optimum می تواند به افزایش سرعت الگوریتم کمک کند. البته می توان احتمال آن را در ابتدا زیاد و در انتهای الگوریتم کم کرد زیرا در آخرای اجرا به جواب های درست همگرا می شویم. mutation می تواند باعث کاهش تعداد رنگ ها نیز بشود.

randomly ↓ 5 → 1

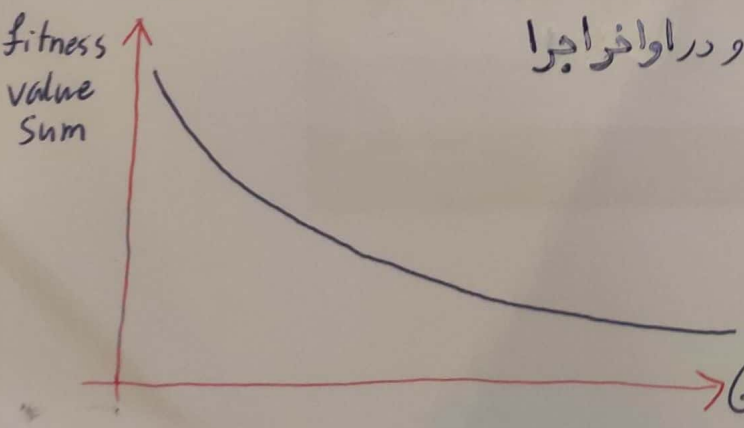
[1, 3, 8, 7, 1, 3, 2, 6, 5, 4, 1, 2, 2, 3] —mutation→ [1, 3, 8, 7, 1, 3, 2, 6, 1, 4, 1, 2, 2, 3]

د) نسل جدید

مجموع Fitness جمعیت فعلی ≥ مجموع Fitness جمعیت قبلی

با گذشت زمان مجموع fitness های هر نسل نسبت به نسل قبل کاهش می یابد و به جواب درست همگرا می شود. با گذشت زمان تعداد رنگ ها کاهش می یابد و به جواب هایی با fitness کمتر و feasible می رسیم. بعد از گذشت چندین نسل دیگر تعداد رنگ ها کمتر نمی شود و الگوریتم می ایستد.

در ابتدا سرعت کاهش مجموع fitness ها بیشتر است و در اواخر اجرا سرعت آن کم می شود.



توجه شود که هرچه fitness کمتر باشد جواب بهتر می شود.

حال الگوریتم ارائه شده در قسمت های بالا را برای یک نسل اجرای کنیم.

الف)

first-generation

1. [1,8,1,2,4,7,5,6,1,8,1,8,3,4,5]
2. [3,3,3,8,6,8,7,2,2,1,2,1,3,4,5]
3. [7,4,5,1,2,7,8,6,6,5,1,2,3,3,8]
4. [1,2,3,3,2,1,8,7,6,6,6,1,1,2,2]
5. [3,4,4,7,8,1,1,2,2,3,5,6,7,8,5]
6. [1,2,2,1,3,3,5,7,5,8,6,7,5,3,4]

node number ← 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

ب)

fitness_value(1) = 1+1+1 = 3

fitness_value(2) = 1+1 = 2

fitness_value(3) = 1 = 1

fitness_value(4) = 1+1+1+1+1 = 5

fitness_value(5) = 1+1+1+1 = 4

fitness_value(6) = 1+1 = 2

مرتب کردن براساس مقادیر fit

3, 2, 6, 1, 5, 4

fitness بیشتر ←

ج) در اینجا از روش Roulette-Wheel Selection استفاده می کنیم.

3, 2, 6, 1, 5, 4

احتمال انتخاب

3: ٪22

2: ٪19

6: ٪19

1: ٪16

5: ٪14

4: ٪10

Selection →

Parents

2
3
2
6
5
1

cross over

نسل جدید

parents

random cross point

2. [3,3,3,8,6,8,7,2,2,1,2,1,3,4,5] → [3,3,3,8,6,8,7,2,2,6,5,1,2,3,3,8]
3. [7,4,5,1,2,7,8,6,6,5,1,2,3,3,8] → [7,4,5,1,2,7,8,6,6,1,2,1,3,4,5]
2. [3,3,3,8,6,8,7,2,2,1,2,1,3,4,5] → [3,3,3,8,6,8,5,7,5,8,6,7,5,3,4]
6. [1,2,2,1,3,3,5,7,5,8,6,7,5,3,4] → [1,2,2,1,3,3,7,2,2,1,2,1,3,4,5]
5. [3,4,4,7,8,2,1,2,2,3,5,6,7,8,5] → [3,4,4,7,8,2,1,6,1,8,1,8,3,3,4,5]
1. [1,8,1,2,4,7,5,6,1,8,1,8,3,4,5] → [1,8,1,2,4,7,5,2,2,3,5,6,7,8,5]

mutation

%20 شانس با احتمال

روی جهو سوم و چهارم انجام می شود.

I. [3,3,3,8,6,8,7,2,2,6,5,1,2,3,3,8]
II. [7,4,5,1,2,7,8,6,6,1,2,1,3,4,5]
III. [3,3,3,8,6,8,5,7,5,8,6,7,5,3,6]
IV. [1,2,2,1,7,3,7,2,2,1,2,1,3,4,5]
V. [3,4,4,7,8,1,1,6,1,8,1,8,3,3,4,5]
VI. [1,8,1,2,4,7,5,2,2,3,5,6,7,8,5]

نسل جدید

second-generation children

fitness_value(I) = 1+1 = 2
fitness_value(II) = 1+1+1 = 3
fitness_value(III) = 1+1 = 2
fitness_value(IV) = 1 = 1
fitness_value(V) = 1+1+1+1 = 4
fitness_value(VI) = 1+1+1 = 3

مجموع fitness جمعیت قبلی = 17

مجموع fitness جمعیت فعلی = 15

یعنی روبه بهبود fitness های کمتر قدم برداشته است.