۱۴۰۱/۱۰/۳۰
چهارمین همایش ملی مهندسی کامپیوتر و نرم افزار
4TH NATIONAL CONFERENCE ON COMPUTER AND SOFTWARE ENGINEERING

حل مسئله رنگ آمیزی گراف با الگوریتم بهینه سازی فاخته

ناهید دلیری بیرجندی
کارشناسی ارشد مهندسی نرم افزار، دانشگاه آزاد اسلامی، واحد بیرجند
پست الکترونیکی : **n.daliri.63@gmail.com**

چکیده

بهینه‌سازی اهمیت زیادی در بسیاری از شاخه‌های علوم همچون فیزیک، شیمی و مهندسی دارد. پژوهشگران در این شاخه‌ها علاقه دارند تا طرح‌های بهینه‌ای برای ایجاد فرآیندهای مختلف به کار برند و میزان تولید محصول را به حداکثر برسانند. الگوریتم های تکاملی مختلف مانند الگوریتم ژنتیک، کلونی زنبور عسل و .. دسته‌ای از الگوریتم‌ها می‌باشد که در حل مسائل بهینه‌سازی در سال‌های اخیر رشد و پیشرفت چشم‌گیری داشته است. الگوریتم بهینه‌سازی فاخته یک روش بهینه‌سازی فرااکتشافی است که رویکردی تکاملی در جستجوی راه‌حل بهینه دارد. این الگوریتم از رفتار جالب توجه گونه‌هایی از پرنده‌ی فاخته در پرورش تخم الهام گرفته است. در این مقاله ابتدا روش زندگی فاخته‌ها و جزییات الگوریتم بهینه‌سازی فاخته مورد بررسی قرار می‌گیرد. سپس به گسسته‌سازی الگوریتم فاخته و تست آن بر روی مسئله رنگ‌آمیزی گراف می بردازیم و در نهایت این الگوریتم با الگوریتم‌های مطرح در این زمینه مقایسه می‌شود.

واژه‌های کلیدی: الگوریتم - الگوریتم بهینه‌سازی فاخته - فضای گسسته - مسئله رنگ‌آمیزی گراف

۱- مقدمه

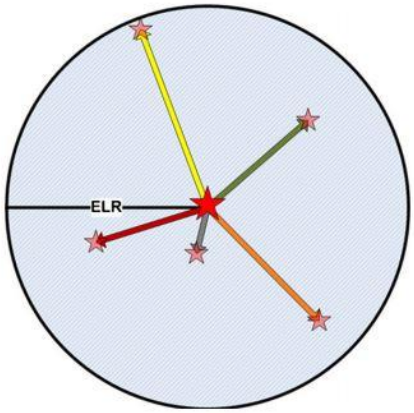
بهینه‌سازی اهمیت زیادی در اکثر شاخه‌های علوم دارد. هدف از این کار، یافتن بهترین پاسخ ممکن برای یک مسئله مشخص است. همین نیاز به جستجو در حل مسائل بهینه‌سازی، منجر به وجود آمدن الگوریتم‌های جستجوی متفاوتی شده است. الگوریتم‌های تکاملی مانند الگوریتم ژنتیک، کلونی زنبور عسل و .. دسته‌ای از این الگوریتم‌ها می‌باشد که در حل مسائل بهینه‌سازی در چند سال اخیر رشد و پیشرفت چشم‌گیری داشته است.[1]

مسائل بهینه‌سازی از دیدگاه‌های متفاوتی طبقه‌بندی می‌شوند، یکی از مهم‌ترین این ابعاد پیوسته و گسسته بودن این مسائل می‌باشد. در این مقاله یکی از جدیدترین الگوریتم‌های تکاملی، الگوریتم بهینه‌سازی فاخته جهت حل مسائل بهینه‌سازی ترکیبی گسسته ارائه می‌شود این الگوریتم از رفتار جالب توجه گونه‌هایی از پرنده‌ی فاخته در پرورش تخم الهام گرفته است و در سال 2009 توسط Yang و Deb پیشنهاد شده است.

۱-۱- روش زندگی و تخم‌گذاری فاخته

فاخته مشهورترین پارازیت اولادی می‌باشد و به نوعی یک متخصص در زمینه فریبکاری است. از عادات یک فاخته حقیقی این است که تخم‌های خود را در یک دامنه مشخص که به آن حداکثر دامنه تخم گذاری ELR گفته می شود می

گذارد. فاخته مادر یکی از تخم‌های پرنده میزبان را که در ELR خود قرار دارد، از بین می برد و تخم خود را در بین تخم‌های دیگر موجود در لانه میزبان قرار می دهد و سریعاً از محل دور می شود. با این عمل، نگهداری از تخم را به عهده پرنده ماده میزبان می گذارد.



شکل (۱) : شعاع تخمگذاری فاخته‌ها

فاخته‌های پارازیت انداز به گروه‌هایی تقسیم می شوند و هر گروه روی پرنده میزبان خاصی تخصص می یابد. البته دراین بین پرندگانی هستند که تخم‌های فاخته‌ها را در لانه‌های خود تشخیص می دهند و حتی تخم‌های فاخته را از لانه خود بیرون می اندازند. بنابراین بعد از هر تخم گذاری p در صد از تمام تخم‌ها (معمولا کمتر از ۱۰ در صد) که مقدار تابع سود آنها کمتر است نابود می شوند. بقیه جوجه‌ها در لانه‌های میزبان تغذیه شده و رشد می کنند.

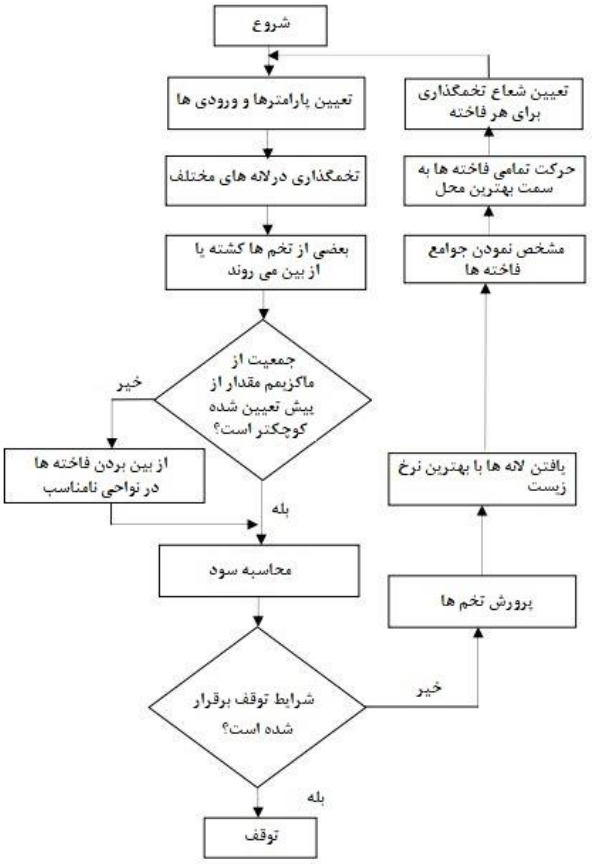
جوجه‌های فاخته زودتر از تخم‌های پرنده میزبان از تخم بیرون می آیند و زودترهم رشد می کنند. در اکثر موارد جوجه ی فاخته تخم‌ها و یا جوجه‌های پرنده میزبان را از لانه بیرون می اندازند. اگر جوجه‌های پرنده میزبان زودتر از تخم خارج شده باشند جوجه فاخته بیشترین مقدار غذا را که پرنده میزبان می آورد خورده و پس از چند روز جوجه‌های خود پرنده از گرسنگی می میرند و فقط جوجه فاخته زنده می ماند.[5-6]

۱-۲- مهاجرت فاخته‌ها

پس از آن که جوجه‌ها از تخم در آمدند و به فاخته بالغ تبدیل شدند، مدتی در گروه‌های خودشان زندگی می‌کنند، ولی وقتی زمان تخمگذاری نزدیک می‌شود به محل سکونت‌های بهتر که در آنجا شانس زنده ماندن تخم‌ها بیشتر است مهاجرت می‌کنند. جهت تشخیص اینکه هر فاخته به کدام گروه تعلق دارد گروه بندی فاخته‌ها توسط روش کلاس بندی K-means انجام می‌شود، سپس سود میانگین گروه محاسبه می‌شود تا بهینگی نسبی محل زیست آن گروه بدست آید. و گروهی که دارای بیشترین مقدار بهینگی باشد، به عنوان گروه هدف انتخاب شده و گروه‌های دیگر به سمت آن مهاجرت می‌کنند. وقتی تمام فاخته‌ها به سمت نقطه هدف مهاجرت کردند و نقاط سکونت جدید هرکدام مشخص شد، هر فاخته صاحب تعدادی تخم می شود. با توجه به تعداد تخم هرفاخته یک ELR برای آن مشخص می‌شود و سپس تخم‌گذاری شروع می‌گردد. [7]

۲- جزئیات الگوریتم بهینه‌سازی فاخته

در شکل (۲) فلوچارت الگوریتم بهینه‌سازی فاخته رسم شده است. همانطور که مشخص است موقعیتی که در آن بیشترین تعداد تخم‌ها نجات یابند پارامتری خواهد بود که الگوریتم بهینه‌سازی فاخته قصد بهینه‌سازی آن را دارد.



شکل (۲) : فلوچارت الگوریتم فاخته[5]

۳- تولید مکان سکونت اولیه و بعدی جستجو

برای حل الگوریتم بهینه‌سازی فاخته لازم است تا مقادیر متغیرهای مساله به فرم آرایه‌هایی که habitat یا "محل سکونت" نام دارند، شکل گیرند. در یک مساله بهینه‌سازی Nvar بعدی یک habitat یک آرایه 1*Nvar خواهد بود که موقعیت فعلی زندگی فاخته‌ها را نشان می دهد. این آرایه به شکل زیر تعریف می شود:

Habitat = [x1,x2,...,xNvar]

مقدار سود در habitat فعلی با ارزیابی تابع سود (fp) در habitat به دست می آید. بنابراین:

Profit=fp(habitat)=fp(x1,x2,...,xNvar)

برای شروع الگوریتم بهینه‌سازی، یک ماتریس habitat به سایز Npop*Nvar تولید می شود. سپس برای هر کدام از این habitatها تعدادی تصادفی تخم تخصیص می یابد. در طبیعت هر فاخته بین ۵ تا ۲۰ تخم می گذارد. این اعداد به عنوان کران-های بالا و پایین تخصیص تخم به هر فاخته در تکرارهای مختلف استفاده می شوند. در یک مساله بهینه‌سازی هر متغیر دارای حد بالا varhi وحد پایین varlow است که هر ELR با استفاده از این حدود قابل تعریف خواهد بود. ELR متناسب با تعداد کل

تخم‌ها، تعداد تخم‌های فعلی فاخته و همچنین حدبالا و پایین متغیرهای مساله است. بنابراین ELR به صورت رابطه (۱) تعریف می شود:

$$ELR = \alpha \times \frac{Numberof\ current cuckod\ s seggs}{Total numberof\ eggs} \times (Var_{hi} - Var_{low}) \qquad (۱)$$

آلفا متغیری است که حداکثر مقدار ELR با آن تنظیم می شود.

پرواز لووی نوعی گشت تصادفی است که طول گام آن از یک توزیع احتمال مشخص تبعیت می‌کند. با استفاده از این روش نقطه‌ی بعدی در جستجو با رابطه‌ی (۲) مشخص می‌شود:

$$L(x) = \frac{1}{\Pi} \int_0^{\infty} \exp(-\alpha q^{\beta}) \cos \quad qx \quad dq \qquad (۲)$$

β اندیس توزیع $(0 < \beta \leq 2)$ و α ضریب مقیاس توزیع $(\alpha > 0)$.

با توجه به این ویژگی می‌توان با بهره‌گیری از پرواز لووی، جستجوی محلی و سراسری را تنها با یک ساختار پیاده‌سازی کرد. بنابراین تعداد پارامترهای مورد نیاز برای اجرای الگوریتم کاهش می‌یابد. [6-8]

۳-۱- همگرایی الگوریتم

پس از چند تکرار تمام جمعیت فاخته‌ها به یک نقطه بهینه با حداکثر شباهت تخم‌ها به تخم‌های پرندگان میزبان و همچنین به محل بیشترین منابع غذایی می‌رسند. این محل بیشترین سود کلی را خواهد داشت و در آن کمترین تعداد تخم‌ها از بین خواهند رفت. همگرایی بیش از ۹۵٪ تمام فاخته‌ها به سمت یک نقطه، الگوریتم بهینه‌سازی فاخته (COA) را به انتهای خود می‌رساند.

۴- الگوریتم فاخته اصلاح شده

این الگوریتم در سال ۲۰۱۲ توسط Humar Kahramanli ارائه شد. در الگوریتم بهینه‌سازی فاخته اصلاح شده روشی جهت کاهش تدریجی شعاع تخم گذاری فاخته‌ها ارائه گردیده است. کاهش مرحله ای شعاع تخم گذاری می تواند منجر به بهبود جستجو شده و دقت جواب‌ها را بالا ببرد. در الگوریتم استاندارد بهینه‌سازی فاخته این تغییر را می توان با کاهش تدریجی ضریب آلفا انجام داد. در روش اصلاح شده شعاع تخم گذاری به صورت رابطه (۳) کاهش می یابد:

$$t = \frac{\max - iter}{c} \qquad (۳)$$

که c یک مقدار ثابت از بازه [۰ ۲۰] و max-iter ماکزیمم تعداد تکرارها است. از اینرو طبق رابطه (۴) داریم:

$$\alpha = \left\| \frac{iteration}{t} \right\| + 1 \qquad e = Var_{Hi} - Var_{Lo} \qquad (۴)$$

که [[ ]] به معنای قسمت صحیح عدد حاصل خواهد بود. بنابراین e در هر تکرار t با استفاده از معادله(۵) اصلاح می شود :

$$e_{new} = \frac{e_{old}}{\alpha} \qquad (۵)$$

از الگوریتم بهینه‌سازی فاخته می‌توان در حل مسائل زیر استفاده کرد:

۱ - مسائل بهینه‌سازی پیچیده با دقت بسیار بالا

۲ - مسائل طراحی چیدمان

۳- مسائل زمانبندی و توالی عملیات

۴- یادگیری شبکه‌های عصبی مصنوعی

و استفاده در هر مساله بهینه‌سازی که قابلیت فرموله شدن بصورت یک تابع هدف در آن وجود دارد.[4]

۵-گسسته‌سازی‌های انجام شده برروی الگوریتم فاخته

با توجه به این که الگوریتم بهینه‌سازی فاخته یک الگوریتم جامع و بسیار سریع می باشد لذا باید قادر باشد تا انواع مسائل بهینه‌سازی را حل نماید.

۵-۱-گسسته‌سازی با استفاده از نگاشت فضا

در این بخش مقداردهی موقعیت‌ها در COA باز تعریف می‌شود که منجر به اعمال الگوریتم پیوسته فاخته در فضای گسسته مسئله می‌شود و نگاشت فضا صورت می‌گیرد. این ایده را از عملیات ریاضی معکوس الهام گرفتیم که تعداد اعداد تصادفی بین حداقل و حداکثر مقدار تولید می‌کند. عملیات معکوس به شرح زیر است:

فرض کنیم Xnorm عددی مابین [0,1] باشد برای تولید به یک عدد در محدوده [minVal,maxVal] داریم:

$$x=(maxVal-minVal) \times X_{norm_+} minVal \quad (۶)$$

این رابطه اعداد مابین [0,1] را به محدوده [minVal,maxVal] نگاشت می‌دهد. اما اعداد تولید شده، ممکن است اعشاری باشد. برای داشتن اعداد صحیح متناظر باید قسمت اعشار را به شرح رابطه (۷) حذف کنیم:

$$floor((maxVal-minVal) \times X_{norm_+} minVal) \quad (۷)$$

همچنین برای تضمین تولید عدد max فرمول را به شکل رابطه(۸) کاملتر کردیم:

$$x=min\{floor((maxVal-minVal) \times X_{norm_+} minVal),maxVal\} \quad (۸)$$

در نهایت با رابطه بالاهر عددی بین [۰ و ۱] به عدد صحیحی در محدوده [minVal,maxVal] نگاشت پیدا می‌کند. حال با این نگاشت می‌توان COA را برای مسائل با فضای گسسته در هر بازه‌ای اعمال کرد. این مدل از گسسته‌سازی را DCOA1 می-نامیم.[2]

۵-۲-گسسته‌سازی با استفاده از تغییر عملگر

جهت اعمال الگوریتم COA به فضای جستجوی گسسته، عملگرهای محاسباتی استاندارد COA نیازمند دوباره تعریف شدن بر روی فضای گسسته می‌باشند که برای انجام این کار مفهوم فاصله و قوانین هندسی برای جواب‌های فضای گسسته معرفی می‌شود و بر مبنای نظریه فاصله عملگرهای COA دوباره تعریف می‌شوند. در ادامه دو مدل گسسته‌سازی برای مسائل جایگشتی و غیرجایگشتی ارائه می‌شود.

۵-۲-۱-مسائل غیربازگشتی

تفاوت مسائلی که از آن به غیرجایگشتی یاد می‌کنیم، با مسائل جایگشتی در این است که در بردارراه‌حل‌های مسائل غیرجایگشتی، اعداد تکراری می‌تواند وجود داشته باشد در حالی که در بردار جایگشتی تکرار اعداد مجاز نیست. از این‌رو تنها تغییر مورد نظر در گسسته‌سازی صورت گرفته برای مسائل غیرجایگشتی، تغییر در عملگر مهاجرت است.

برای تعمیم COA به فضای جستجوی گسسته، نیازمند دوباره تعریف کردن تمام عملگرهای محاسباتی در عملگر مهاجرت هستیم که مهم‌ترین آن‌ها اختلاف موقعیت دو محل سکونت می‌باشد. مفهوم اصلی مهاجرت در COA به این صورت است که هر فاخته سعی دارد خود را به بهترین نقطه هدف نزدیک‌تر سازد. بر طبق تعریف فاصله روی فضای جستجوی گسسته، برای هر موقعیت x و y، x می‌تواند از طریق اعمال عملگر که کوتاه‌ترین مسیر بین آنها باشد به y تبدیل شود. هر اعمالی از عملگر فاصله بین x و y را یک واحد کاهش می‌دهد. بنابراین فاصله دو موقعیت از فاخته‌ها به عنوان دنباله‌ای از تعداد اعمال عملگر تعریف می‌شود که یک موقعیت را به دیگری تبدیل می‌کند.

در فاز مهاجرت از COA، هدف حرکت فاخته‌ها به سمت بهترین جمعیت است. ابتدا ما رابطه مهاجرت در COA پیوسته را که در رابطه (۹) نشان داده شده است، تغییر خواهیم داد. طوری که هر دو ویژگی مهاجرت در COA را پشتیبانی کند.

$$X_{NextHabitat} = X_{CurrentHabitat} + F \times (X_{GoalPoint} - X_{CurrentHabitat}) \quad (۹)$$

F یک عدد تصادفی است که گام‌های حرکتی به سمت نقطه هدف را تعیین می‌کند. بنابراین در COA گسسته روشی برای مدل کردن این حرکت و رسیدن به نقطه هدف در هر تکرار مورد نیاز است. در ادامه مفاهیم تفریق، ضرب و جمع برای فضای گسسته باز تعریف می‌شود.

اختلاف بین دو موقعیت: برای هر موقعیت x و y اختلاف آنها (x-y) برابر است با دنباله‌ای از حداقل تعداد دفعات به کار بردن عملگر است به این معنی

$$x\text{-}y=M=O(x),O(r),\ldots,O(t) \quad (۱۰)$$

بنابراین

$$O(x)=r,\ O(r)=s,\ldots,O(t)=y, \quad (۱۱)$$
$$x,r,s,\ldots,t,y\ [x,y]\in$$

به این معنی که عملگر اختلاف در الگوریتم فاخته برای مسائل غیرجایگشتی برابر است با توالی تعداد دفعات به کار رفته عملگر تغییر عناصر می‌باشد. به عبارتی برای به دست آوردن اختلاف دو بردار در فضای گسسته، لیست حرکات ضروری برای تغییر از راه‌حل S_j به S_i تحت عنوان $M_{j\to i}$ که از طریق تغییر عناصر به دست می‌آید را تعریف کردیم. اختلاف بین دو راه‌حل به صورت رابطه (۱۲) نشان داده شده است:

$$M_{j\to i} = S_i \boxminus S_j \quad (۱۲)$$

رابطه (6) به این معنی است که اگر حرکات در لیست $M_{j\to i}$ روی راه‌حل S_j اعمال شود راه‌حل S_i به دست خواهد آمد. لیست $M_{j\to i}$ اختلاف بین نقطه هدف و موقعیت جاری فاخته‌ها را مدل می‌کند. به این صورت اختلاف دو نقطه برابر است با لیست حرکاتی که با عبارت سه مؤلفه‌ای (a,b,c) نمایش داده می‌شود به این معنی که در اندیس aی بردار، عدد b را به c تغییر دهد.

ضرب: با در نظر گرفتن $F \in [0,1]$ که یک عدد اعشاری است و $M' = F \times |M|$، در اینجا عملگر ضرب به این معنی می‌باشد که تعدادی از حرکات موجود در لیست M انتخاب می‌شود. در واقع برای تکمیل کردن عملگر، از پارامتر F به عنوان یک گام تصادفی به سمت نقطه هدف تعریف شده استفاده می‌شود.

جمع: اگر X موقعیت و M' لیست حرکات باشد، $x + M'$ به معنی اعمال لیست حرکات M' روی بردار X می‌باشد که یک موقعیت جدید در فضا را ایجاد می‌کند.

بنابراین عملگر مهاجرت به فرم (۱۳) تغییر می‌یابد:

$$M_{j \to i} = X_{GoalPoint} \boxminus X_{CurrentHabitat}$$
$$X_{NextHabitat} = X_{CurrentHabitat} \oplus F \otimes M_{j \to i} \quad (۱۳)$$

این مدل از گسسته‌سازی برای مسائل غیرجایگشتی که DCOA2 نامیدیم .[9]

۵-۲-۲- مسائل جایگشتی

استفاده از عملگرها و تعاریف جدید از تخم‌گذاری و مهاجرت در COA، امکان حل مسائل جایگشتی را فراهم می‌آورد. برای گسسته‌سازی این مسائل نیاز به دو تغییر در الگوریتم فاخته احساس می‌شود. اول این‌که پیشنهاد می‌شود که تخم‌گذاری در الگوریتم بهینه‌سازی فاخته را تغییر داد تا تخم‌ها در موقعیت مناسب قرار گیرد و تغییر دوم باز تعریف عملگر مهاجرت می‌باشد که در ادامه شرح داده می‌شود.[8]

- تعریف تخم گذاری برای مسائل جایگشتی

سه عملگر متفاوت در این بخش برای تولید موقعیت‌های مناسب برای تخم‌گذاری جدید تعریف می‌شود که این روش‌ها الهام گرفته از عملگر جهش در الگوریتم ژنتیک می‌باشند . در هر تکرار از الگوریتم بهینه‌سازی یکی از این سه عملگر به طور تصادفی انتخاب و برای تعیین موقعیت جدید فرزندان اعمال می‌شود.

روش ۱:

اگر [x1, x2, x3, x4, x5, x6] نمونه‌ای از موقعیت یک فاخته باشد. دو نقطه تصادفی برای جایگزینی به صورت شکل۳) انتخاب می‌شود:

x1	x2	x3	x4	x5	x6

شکل(۳): نمایش بردار x با بیت‌های انتخاب شده[۲]

سپس با جایگزینی این دو نقطه به صورت شکل (۴)، یک موقعیت جدید برای تخم‌گذاری که گذاشته می‌شود، تولید می‌شود.

x1	x5	x3	x4	x2	x6

شکل (۴): اعمال جهش روش ۱ بر روی بردارx [۲]

روش ۲:

در این روش، چند موقعیت ترتیبی به صورت شکل(۵) انتخاب می‌شود:

x1	x2	x3	x4	x5	x6

شکل (۵): نمایش بردار x با بیت‌های انتخاب شده[۲]

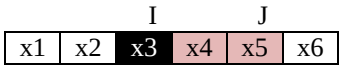
با اعمال روش دوم تمام این بیت‌های انتخابی از چپ به راست به صورت شکل (۶) معکوس می‌شوند.

x1	x5	x4	x3	x2	x6

شکل (۶): اعمال جهش روش ۲ بر روی بردارx [۲]

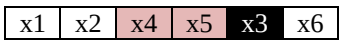
روش ۳:

چند بیت بین دو موقعیت I و J به صورت شکل (۷) انتخاب می‌شود:



شکل (۷): نمایش بردار x با بیت‌های انتخاب شده[۲]

سپس یک جایگزینی به صورت شکل(۸) انجام می‌گردد که محتوای موقعیت I+1 تا J به موقعیت I شیفت داده می‌شود و محتوای موقعیت I به دنبال آنها قرار می‌گیرد.



شکل (۸): اعمال جهش روش ۳ بر روی بردارx [۲]

بعد از این‌که موقعیت‌های جدید برای تخم‌ها به وسیله‌ی یکی از این سه روش تعیین شد، تخم‌ها گذاشته شده و سپس شایستگی موقعیت‌ها محاسبه می‌شود.

- باز تعریف عملگر مهاجرت

در این بخش از تغییر عملگر مهاجرت تمام تعاریف مانند بخش عملگر مهاجرت DCOA2 می‌باشد با این تفاوت که در محاسبه اختلاف دو بردار برای مسائل جایگشتی، به جای استفاده از عملگر تغییر عناصر، عملگر جایگزینی را تعریف کردیم. لیست حرکات با عبارت دو مؤلفه‌ای (a,b) نمایش داده می‌شود به این معنی که محتویات موجود در اندیس a را با محتویات موجود در اندیس b جایگزین نماید.

۶- حل مسئله رنگ آمیزی گراف

مسئله رنگ‌آمیزی گراف یکی از مسائل شناخته شده در زمینه بهینه‌سازی ترکیبی می‌باشد. فرض می‌کنیم گراف G=(V, E) یک گراف فاقد حلقه و V مجموعه رئوس و E مجموعه یال‌های گراف باشند. در مسئله رنگ‌آمیزی گراف می‌خواهیم همه گره‌ها را به گونه‌ای رنگ‌آمیزی کنیم که هیچ دو گره مجاوری هم‌رنگ نباشند، ضمن اینکه کمترین تعداد رنگ را به کار برده باشیم. [10-11]

۶-۱-حل مساله رنگ‌آمیزی گراف با الگوریتم DCOA1

برای رنگ‌آمیزی گراف به وسیله‌ی این الگوریتم، کافی است تابع هزینه‌ای برای مسئله طراحی نماییم و در همان ابتدا تعداد رنگ شناخته شده برای هر گراف را به عنوان ابعاد مورد بهینه‌سازی به عنوان ورودی به الگوریتم DCOA1 می‌دهیم. هدف به دست آوردن تعداد رنگ‌ها بدون برخورد می‌باشد. الگوریتم DCOA1 اعمال شده بر روی مسئله رنگ‌آمیزی گراف را DCOACOL1 نامیدیم. تابع هزینه تعریف شده ما شامل ۳ بخش مختلف است و به شرح زیر می‌باشد:

اگر یالی بین گره i وj وجود دارد و رنگ پیشنهاد شده برای ۲ گره یکسان باشد، conflict یک واحد افزایش می‌یابد که conflict به صورت رابطه (۱۴) محاسبه می‌شود:

$$f(x) = \sum_{\{i,\, j\} \in E} conflict_{ij} \qquad (۱۴)$$

$$conflict_{ij} = \begin{cases} 1 & if\ i\ and\ j\ have\ the\ same\ color \\ 0 & otherwise \end{cases}$$

- اگر یالی بین گره i وj وجود نداشته باشد و رنگ پیشنهاد شده برای ۲ گره یکسان نباشد، penalty یک واحد افزایش می‌یابد. این امر برای کاهش تعداد رنگ‌های مورد استفاده به کار می‌رود.

۶-۲-حل مسئله رنگ‌آمیزی گراف با استفاده از الگوریتم DCOA2

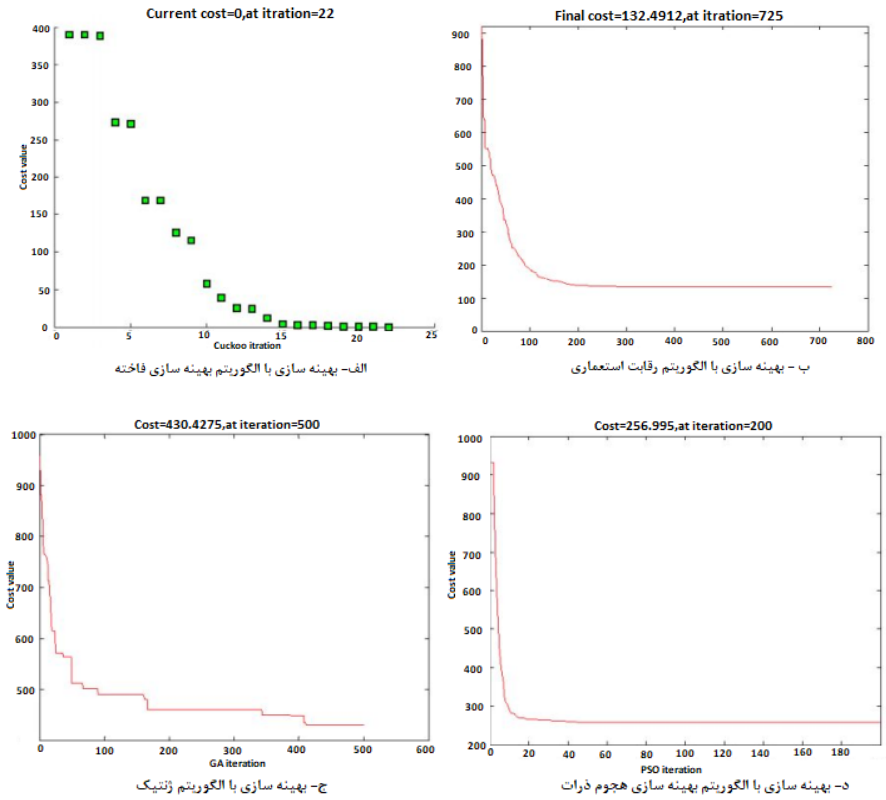
گراف با تعداد ثابت K به عنوان ورودی به الگوریتم داده می‌شود و خروجی مسئله پیدا کردن k رنگ‌آمیزی مناسب است. در ابتدا K=|V| مقداردهی می‌شود. ابتدا یک گره به صورت تصادفی انتخاب شده و توسط الگوریتم Coloring Vertex رنگ می شود. سپس گره‌های دیگر بر اساس درجه‌ای که دارند مرتب شده و با توجه به رنگ گره اول، به ترتیب هر کدام با الگوریتم Coloring Vertex رنگ خاصی به خود می‌گیرند. برای شروع دارای جواب‌هایی هستیم که هیچ تداخل رنگی‌ایی با هم ندارند. حال فقط باید COA اقدام به کاهش تعداد رنگ‌های مورد استفاده نماید. برای این‌کار جستجوی همسایگی در بخش تخم گذاری الگوریتم فاخته به منظور حداقل کردن تعداد رنگ‌ها به طوری که برخورد پیش نیاید، اعمال می‌شود . در این بخش از الگوریتم COA با توجه به موقعیت جمعیت فعلی فاخته‌ها، اقدام به تولید جایگشت‌های بهینه‌ای برای تخم‌های فاخته می‌کنیم .

۷-برتری‌های الگوریتم بهینه‌سازی فاخته

با توجه به اینکه هر ساله الگوریتم‌های جدیدتری با پوشاندن نقاط ضعف روش‌های قبلی پا به عرصه ظهور می گذارند نمیتوان ادعا کرد یک الگوریتم بهینه‌سازی بهترین روش حل برای مسائل بهینه‌سازی است. در کاربردهای تست شده تاکنون، الگوریتم بهینه‌سازی فاخته بسیار بهتر از بقیه الگوریتم‌ها عمل کرده است.

دلیل این که الگوریتم بهینه‌سازی فاخته بهتر از سایر الگوریتم‌های تکاملی عمل می کند در کارکرد چندگانه عملگرهای الگوریتم بهینه‌سازی فاخته، مثل تخمگذاری و مهاجرت مستتر می باشد. در سایر الگوریتم‌های بهینه‌سازی تکاملی با عملگرهایی مواجه می شویم که فقط یک هدف خاص را بر عهده دارند. ولی در الگوریتم بهینه‌سازی فاخته عملگرهای تعریف شده به صورت همزمان چندین هدف را تحقق می بخشند.

الگوریتم بهینه‌سازی فاخته به دلیل الگوریتم خاص و اصلاح یافته ای که دارد و در آن بیشتر مشکلات و ضعف‌های الگوریتم های بهینه‌سازی تکاملی قبلی از الگوریتم ژنتیک گرفته تا الگوریتم رقابت استعماری که نسبتاً جدیدتر می باشد، به نوعی رفع شده و بنابراین دارای توانایی همگرایی بسیار سریعتر و قدرت یافتن نقاط بهینه کلی به صورت بسیار دقیق تری می باشد. الگوریتم بهینه‌سازی فاخته قادر است با ترکیب چندین عملگر که کمک شایانی به جستجوی محلی در حین جستجوی کلی می کنند به جواب‌های بسیار دقیق تری دست یابد. به عنوان مثال شکل (۹) مربوط به پیدا کردن نقطه بهینه تابع بسیار پیچیده ی راستریگین با ۱۰۰ بعد می باشد. نتایج نشان می دهد در جایی که الگوریتم‌های دیگر حتی با جمعیت بیشتر، از نزدیک شدن به جواب بهینه عاجزند الگوریتم بهینه‌سازی فاخته به راحتی و آن هم در تعداد تکرارهای بسیار کمی توانسته جواب بهینه کلی را با دقت ۱۰۰ درصد به دست آورد و تابع هزینه را دقیقاً صفر کند . [3-12]



شکل (۹): مقایسه الگوریتم فاخته با الگوریتمهای مشابه

۸-نتیجه

در این مقاله، الگوریتم بهینه‌سازی که از شیوه زندگی پرنده ای به نام فاخته الهام گرفته شده بود، ارائه گشت. ویژگی-های خاص cuckoos در تخم گذاری و پرورش جوجه‌ها، انگیزه اساسی برای توسعه این الگوریتم جدید بهینه‌سازی شده است. با توجه به اینکه مسئله رنگ‌آمیزی گراف از جمله مسائلی است که در فضای گسسته قابل حل می‌باشد به تست الگوریتم فاخته بر روی مسئله رنگ‌آمیزی گراف پرداختیم . مقایسه این الگوریتم با نسخه‌های استاندارد PSO و GA با انتخاب، برتری COA در همگرایی سریع و دستیابی به OPTIMAجهانی را نشان داد.

هرجند لزوما به این معنی نیست که COA همیشه بهترین روش تکاملی توسعه یافته است. این الگوریتم تنها می تواند برای نوعی از مشکلات بهینه‌سازی به عنوان تقلید موفق از طبیعت در نظر گرفته شود .

مراجع

[1] J.Qin, X.Xu, Q.Wu,T.C.E. Cheng, Hybridization of tabu search with feasible and infeasible local searches for the quadratic multiple knapsack problem, Computers & Operations Research.66(2016) 199–214.

[2] M. Aghaie, S.M. Mahmoudi, A novel multi objective Loading Pattern Optimization by Gravitational Search Algorithm (GSA) for WWER1000 core, Progress in Nuclear Energy.93(2016)1-11.

[3] I. Fister Jr, X. Yang, D. Fister and I. Fister. (2014), "Cuckoo Search: A Brief Literature Review", Cuckoo Search and Firefly Algorithm: Theory and Applications, Studies in Computational Intelligence, vol. 516, pp. 49-62.
[4] H. Kahramanli. (2012), "A Modified Cuckoo Optimization Algorithm for Engineering Optimization", International Journal of Future Computer and Communication Vol 1(2), pp. 199-201.
[5] S. Mahmoudi, S. Lotfi, Modified cuckoo optimization algorithm (MCOA) to solve graph coloring problem, Appl. Soft Comput. 33 (2015) 48–64.
[6] R. JOVANOVIC, M. TUBA and M. BRAJEVIC. (2013), "Parallelization of the Cuckoo Search Using CUDA Architecture", Ministry of education and science of Republic of Serbia, Grants III-44006 and 171006.
[7] Kahramanli, H, "*A Modified Cuckoo Optimization Algorithm for Engineering Optimization", International Journal of Future Computer and Communication*, Vol 1(2), pp 199-201.
[8] Rajabioun, R., *Cuckoo Optimization Algorithm, In: Applied Soft Computing journal,* Vol. 11, pp. 5508-5518, 2011.
[9] Yang, X.S. & Deb, S., "*Cuckoo search via Lévy Flights", In: World Congress on Nature & Biologically Inspired Computing (NaBIC 2009).* IEEE Publications, pp. 210-214, 2009.
[10] L. Huang, Sh.Ding, Sh. Yu, J. Wang,Chaos-enhanced Cuckoo search optimization algorithms for global optimization, Applied Mathematical Modelling.40(2016) 3860– 3875.
[11] Faraji, M. and Haj Seyyed Javadi, H., "Proposing a New Algorithm Based on Bees Behavior for Solving Graph.
[12] Coloring", Int. J. Contemp. Math. Sciences, Vol. 6, No. 1, pp. 41 - 49, 2011.