**چکیده**

**در بهینه سازی به دنبال یافتن بهترین مقدار قابل دستیابی از یک تابع هدف تعریف شده بر یک دامنه معین از مقادیر هستیم.** **بهینه سازی در طراحی و حفظ بسیاری از سیستم های مهندسی، اقتصادی و حتی اجتماعی با هدف کمینه نمودن هزینه یا بیشینه نمودن سود مورد استفاده قرار می گیرد. در سال های اخیر به علت کاربردهای وسیع بهینه سازی در علوم مختلف، توجه زیادی به این مبحث گردیده است. در این پایان نامه یک روش بهینه سازی فرا ابتکاری ارائه شد که از نحوه زندگی جاندارانی با نام کوآلا الهام گرفته شده است. نتایج حاصل از اعمال الگوریتم پیشنهادی بر توابع معیار سنجش در نظر گرفته شده در کارهای مشابه، حاکی از بهبود عملکرد بهینه سازی این الگوریتم است.**

**کلمات کلیدی: بهینه سازی[[1]](#footnote-1)، الگوریتم فراابتکاری[[2]](#footnote-2)، کوآلا[[3]](#footnote-3)، کمینه سازی[[4]](#footnote-4)، بیشینه سازی[[5]](#footnote-5)**

**فصل اول**

**مقدمه**

**بهینه سازی فرآیندی است که در آن نتیجه را تا حد امکان کامل تر، کارآمدتر و موثرتر می نماییم. درواقع بهینه سازی فرآیند بهتر نمودن چیزی است]1[. هر سیستم بهینه سازی شامل ورودی، تابع ارزیاب[[6]](#footnote-6) که براساس آن میزان خوب بودن سیستم را می سنجند و از آن با نام های تابع هزینه[[7]](#footnote-7) یا تابع هدف[[8]](#footnote-8) نیز یاد می شود و خروجی که شامل سود یا هزینه است می شود. بهینه سازی یک فرآیند ریاضی پرکاربرد در تمامی رشته های مهندسی است که در آن به دنبال پیدا کردن راه حل مطلوب هستیم. از آنجایی که دامنه مسائل بهینه سازی متنوع است، روش های بهینه سازی متناسب با آنها نیز از تنوع زیادی برخوردار است.**

**به طورکلی دو روش بهینه سازی کلاسیک[[9]](#footnote-9) و بهینه سازی هوش مصنوعی[[10]](#footnote-10) وجود دارد. بهینه سازی کلاسیک محاسبات لازم برای یافتن نقاط بهینه در یک تابع پیوسته[[11]](#footnote-11) است. این روش ها تحلیلی می باشند و برای شناسایی نقاط بهینه توابع از محاسبات دیفرانسیل[[12]](#footnote-12) استفاده می نمایند]4[.انواع الگوریتم های بهینه سازی کلاسیک شامل بهینه سازی توابع تک متغیره[[13]](#footnote-13)، توابع چند متغیره بدون محدودیت[[14]](#footnote-14) و توابع چند متغیره با محدودیت[[15]](#footnote-15) می شود]5[.**

**الگوریتم های کلاسیک با وجود توانایی شناسایی نقطه بهینه دقیق، به علت وابسته بودن زمان اجرای آن با ابعاد مساله، درمورد مسائل سخت و پیچیده و با ابعاد بالا گزینه مناسبی نمی باشند.**

**الگوریتم های بهینه سازی در هوش مصنوعی در زمان کمتر و قابل قبول تری قادر به یافتن جوابی نزدیک به جواب بهینه هستند. این الگوریتم ها شامل الگوریتم های بهینه سازی ابتکاری[[16]](#footnote-16)، فراابتکاری[[17]](#footnote-17) و فوق فراابتکاری[[18]](#footnote-18) می شوند.**

**به طور کلی الگوریتم های ابتکاری مساله محور هستند و برای یک مساله خاص طراحی می شوند. لذا از این الگوریتم ها برای پاسخ به مسائل مختلف نمی توان بهره برد. ضمن اینکه این الگوریتم ها معمولا در یک جواب بهینه محلی گرفتار می شوند.در نقطه مقابل الگوریتم های فراابتکاری مساله محور نبوده و برای پاسخ دادن به مسائل متنوع می توانند استفاده شوند. همچنین این الگوریتم ها بااستفاده از روش هایی سعی می کنند در نقطه بهینه محلی گرفتار نشوند.**

**الگوریتم های فراابتکاری براساس معیارهای مختلف دارای دسته بندی های مختلفی است. یکی از این دسته بندی ها، دسته بندی الگوریتم های فراابتکاری به دو دسته الگوریتم های فراابتکاری الهام گرفته شده از طبیعت و بدون الهام از طبیعت است]6[.**

**طبیعت به دلیل پویایی، متنوع بودن مسائل، مقاوم بودن در مقابل بسیاری از عوامل، پیچیدگی وجذاب بودن، یک منبع بسیار خوب برای الهام گرفتن روش های حل مسائل سخت و پیچیده است. امروزه محاسبات الهام گرفته شده از طبیعت توجه بسیاری از محققان را در علوم رایانه، هوش محاسباتی و بهینه سازی به خود جلب نموده است.**

**الگوریتم های فراابتکاری الهام گرفته شده از طبیعت، با یک جمعیت اولیه تصادفی شروع به کار نموده و سپس در طول یک روند تکراری سعی در بهبود پارامترها می نمایند. الگوریتم های ژنتیک، کلونی مورچگان، زنبور عسل، ازدحام ذرات و فاخته ، مثال هایی از این دسته الگوریتم های می باشند.**

**از جمله مزایای این الگوریتم ها می توان به موارد زیر اشاره نمود]1[:**

* **کاربرد گسترده آن در زمینه های مختلف**
* **قابلیت ترکیب با الگوریتم های دیگر**
* **توانایی حل مسائلی که هیچ راه حلی ندارند**
  1. **اهمیت مساله و ضرورت انجام**

**بسیاری از اعمال انسان، بازدهی صد درصد ندارد و به نتیجه مطلوب نمی رسد، از این رو استفاده بهینه و ممانعت از هدر رفتن امكانات و یافتن بهترین و بهینه ترین راه انجام این اعمال، امری اساسی است.**

**در سال‌های اخیر به علت کاربردهای وسیع بهینه‌سازی در علوم مختلف، توجه زیادی به این مبحث گردیده و درنتیجه شاهد رشد چشمگیری در این زمینه بوده‌ایم. عموماً برای مقایسه الگوریتم‌های بهینه‌سازی دو معیار همگرایی[[19]](#footnote-19) و عملكرد[[20]](#footnote-20) در نظر گرفته می‌شود]3[. بعضی از الگوریتم‌ها دارای همگرایی مطلوبی بوده اما ممكن است عملكرد ضعیفی داشته باشند، یعنی فرایند بهبود آن‌ها از كارآیی و سرعت لازم برخوردار نباشد، بالعکس بعضی دیگر از الگوریتم‌ها همگرایی مطلوبی ندارند اما عملكرد آن‌ها بسیار خوب است**.

**سرعت بالای پیشرفت علم و در پی آن مطرح شدن مسائلی با پیچیدگی بیشتر و همچنین نیاز انسان برای رسیدن هرچه سریع‌تر به بهترین پاسخ باعث شده است که باوجود الگوریتم‌های مختلفی که تاکنون برای بهینه‌سازی مطرح شده است، همچنان نیاز به الگوریتمی با سرعت و عملکرد بهتر احساس شود.**

* 1. **هدف پژوهش**

**در واقع هدف این پایان نامه ارائه یک روش بهینه سازی نوین بر اساس نحوه زندگی کوآلا است که نسبت به الگوریتم فاخته دارای سرعت و دقت بیشتری در حل مسائل بهینه سازی است.**

* 1. **روش پیشنهادی**

**در این روش مانند تمامی الگوریتم های فراابتکاری دیگر با یک جمعیت اولیه تصادفی شروع به کار نموده و سپس عمل ترکیب برای تولید فرزندان که درواقع جمعیت جدید است، صورت می گیرد . پس از آن والد شروع به بهبود شرایط خود نموده و پس از بهبودی خود، فرزندان را در جهت خود بهبود می دهد.**

* 1. **ساختار پایان نامه**

**مطالب عنوان شده در این پایان نامه در قالب پنج فصل آورده شده است. ادامه مطالب ذکر شده به شرح زیر می باشد:**

**در فصل دوم به توضیح روش ها و الگوریتم های بهینه سازی هوش مصنوعی پرداخته ایم.**

**در فصل سوم به شرح الگوریتم پیشنهادی و ارائه فلوچارت آن اختصاص یافته است.**

**در فصل چهارم عملکرد روش پیشنهادی را با روش های بهینه سازی موجود مقایسه و نتایج ارائه شده است.**

**در فصل پنجم نتیجه گیری آورده شده است.**

**فصل دوم**

**پیشینه پژوهشی**

**در این فصل به بررسی کارهای انجام شده در گذشته در زمینه بهینه سازی هوش مصنوعی پرداخته می شود. الگوریتم هایی که در این فصل مورد بررسی قرار گرفته اند عبارتند از : الگوریتم ژنتیک[[21]](#footnote-21)، الگوریتم فاخته[[22]](#footnote-22)، الگوریتم کلونی مورچگان[[23]](#footnote-23)، الگوریتم زنبورها[[24]](#footnote-24)، الگوریتم ازدحام ذرات[[25]](#footnote-25)و الگوریتم تبرید شبیه سازی شده[[26]](#footnote-26).**

**2-1- الگوریتم ژنتیک**

**الگوریتم ژنتیک یک الگوریتم بهینه سازی تکاملی تصادفی است که توسط جان هلند[[27]](#footnote-27) ارائه شده است و پتانسیل جستجوی سراسری[[28]](#footnote-28) را دارد]12[. ایده اصلی این الگوریتم از اصول نظریه انتخاب طبیعی[[29]](#footnote-29) گرفته شده است. عملگرهای اصلی الگوریتم ژنتیک برای ایجاد نسل جدید عبارتند از ترکیب[[30]](#footnote-30) و جهش[[31]](#footnote-31). الگوریتم با یک جمعیت اولیه تصادفی از راه حل ها شروع به کار می نماید. سپس برای هر یک از آنها تابع ارزیاب[[32]](#footnote-32) مناسب مساله محاسبه می شود. براساس این ارزیابی بهترین ها برای ترکیب و جهش انتخاب می شوند تا نسل بعدی را به وجود بیاورند.**

**2-1-1- کاربردهای الگوریتم ژنتیک**

**الگوریتم ژنتیک در مسائل زیر بسیار مفید و موثر است]2[:**

* **مسائل با فضای جستجوی بزرگ، پیچیده و ناشناخته**
* **مسائلی که هیچ تحلیل ریاضیاتی برای آنها موجود نباشد.**
* **مسائلی که درآنها دامنه دانش برای رمزگذاری به منظور محدود نمودن فضای جستجو کم باشد.**
* **مسائلی که روش های جستجوی سنتی[[33]](#footnote-33) در آنها شکست خورده است.**

**2-1-2- شبه کد الگوریتم ژنتیک**

1. **ایجاد جمعیت اولیه**
2. **محاسبه تابع ارزیاب مساله برای جمعیت**
3. **تا رسیدن به شرط خاتمه مراحل زیر تکرار شود:**
4. **انتخاب والدین مناسب برای تولید مثل**
5. **اعمال عملگر ترکیب**
6. **اعمال عملگر جهش**
7. **انتخاب جمعیت جدید از میان جمعیت موجود**

**2-1-3- عملگر ترکیب**

**این عملگر توسط الگوریتم ژنتیک، به منظور تولید نسل جدید، از والدین با قدرت و توانایی مطلوب صورت می گیرد. روش های گوناگونی برای این عملگر وجود دارد. ازجمله ترکیب تک نقطه ای[[34]](#footnote-34)، ترکیب چند نقطه ای[[35]](#footnote-35) و ترکیب یکنواخت[[36]](#footnote-36) و غیره .**

**2-1-3-1- ترکیب تک نقطه ای**

**در این روش ابتدا یک نقطه بر روی دو والد برگزیده شده، انتخاب می شود. سپس اطلاعات قبل و بعد این نقطه میان دو والد جابجا می شوند. نتیجه این فرآیند به شکل زیر می باشد]33[.**

**شکل 2-1 ترکیب تک نقطه ای**

**2-1-3-2- ترکیب دو نقطه ای**

**در ترکیب دو نقطه ای، همانطور که از نام آن پیداست، دو نقطه را انتخاب کرده و اطلاعات میان آن دو نقزه را در والدین جابجا می نماییم.**

**شکل2-2 ترکیب دو نقطه ای**

**2-1-3-3- ترکیب یکنواخت**

**در این روش برخلاف دو روش بیان شده، به جای انتخاب نقاط خاصی برای ترکیب، به هر عنصر والدین یک عدد تصادفی 0 یا 1 اختصاص می دهیم. در صورت یک بودن، مقدار عنصر مربوطه، در فرزند ایجاد شده قرار خواهد گرفت]32[. برای فهم بهتر می توان از شکل زیر کمک گرفت.**

**شکل 2-3 ترکیب یکنواخت**

**2-1-4- عملگر جهش**

**عملگر جهش عملگر دیگری از الگوریتم ژنتیک است که جستجوی اکتشافی مساله را انجام می دهد. این عملگر به روش های مختلفی قابل پیاده سازی است از جمله به روش باینری، تعویضی، وارونه سازی، درجی، تغییر مقدارو... . در این بخش به شرح مختصری از این روش ها می پردازیم.**

**2-1-4-1- جهش باینری**

**در این روش با توجه به نرخ جهش، برخی آلل ها متمم می گردند.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **0** | **1** | **1** | **0** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **1** | **0** | **1** | **1** | **1** |

**شکل 2-4 جهش باینری**

**2-1-4-2- جهش تعویضی[[37]](#footnote-37)**

**در این روش دو آلل به صورت تصادفی انتخاب و باهم جابجا می شوند.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **2** | **5** | **4** | **3** | **6** | **1** |

**شکل2-5 جهش تعویضی**

**2-1-4-3- جهش وارونه سازی[[38]](#footnote-38)**

**در این روش دو آلل انتخاب شده و مقادیر ژن های بین آنها وارونه می گردد.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **2** | **3** | **4** | **5** | **6** | **1** |

**شکل 2-6 جهش وارونه سازی**

**2-1-4-4- جهش درجی**

**در این روش دو آلل را به صورت تصادفی انتخاب نموده و آلل دوم را به آلل بعد از اولی انتقال داده و باقی آلل ها شیفت داده می شود.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **4** | **3** | **5** | **2** | **1** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** |

**شکل 2-7 جهش درجی**

**2-1-4-5- جهش تغییر مقدار**

**این روش برای کروموزوم هایی که به صورت باینری کدگذاری شده اند قابل اعمال نیست. در این روش با توجه به نرخ جهش از مقدار برخی آلل ها، عددی کاسته یا به آن افزوده می گردد.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** | **1** | **5** | **4** | **2** | **2** | **1** |

**شکل 2-8 جهش تغییر مفدار**

**2-1-5- شرایط خاتمه**

**الگوریتم براساس مساله و فضای جستجوی آن و نظر طراح می تواند یک یا بیش از یک شرط خاتمه داشته باشد که در صورت دست یافتن به آنها متوقف شود. از جمله این شرایط می توان به موارد زیر اشاره نمود:**

* **رسیدن به تعداد ثابتی تکرار**
* **محدودیت زمان محاسبه یا هزینه**
* **برآورده شدن تابع ارزیاب**
* **ثابت ماندن و یا نرسیدن به نتایج بهتر**

**2-2- الگوریتم کلونی مورچگان**

**الگوریتم کلونی مورچگان]16[در سال 1996 توسط دوریگو[[39]](#footnote-39) ارائه شد. این الگوریتم یک متاهیوریستیک جمعیت محور است که قادر است راه حل های تقریبی[[40]](#footnote-40) را برای مسائل بهینه سازی سخت بیابد]15[. روش کار این الگوریتم براساس روش کارکرد مستعمرات مورچه است که یک رویکرد قابل اجرا برای بهینه سازی ترکیبی تصادفی است.**

**2-2-1- نحوه زندگی مورچگان**

**مطالعات نشان داده است که مورچه ها حشراتی اجتماعی هستند که در کلونی ها زندگی می کنند و رفتار آنها بیشتر جهت بقاء کلونی است تا در جهت بقاء یک جزء از آن. یکی از مهمترین و جالب ترین رفتار مورچه ها، رفتار آنها برای یافتن غذا و چگونگی پیدا کردن کوتاه ترین مسیر میان منابع غذایی و آشیانه است. این نوع رفتار مورچه ها دارای نوعی هوشمندی توده ای است که اخیرا مورد توجه دانشمندان قرار گرفته است. با وجود اینکه مورچه ها موجوداتی کم هوش و بی حافظه هستند، همیشه بهینه ترین مسیر را برای یافتن غذا پیدا می کنند. ارتباط مورچه ها با یکدیگر از طریق ماده شیمیایی با نام فرومون[[41]](#footnote-41) است که مورچه در طول مسیر خود ترشح می نماید و از این طریق مورچه های دیگر به دنبال او حرکت می کنند. فومون به جا گذاشته در مسیر با گذشت زمان تبخیر می شود و غلظت آن کاهش می یابد در نتیجه مسیرهایی که به تازگی کشف شده اند دارای فرومون بیشتر می باشند. می‌توان این غلظت فرومون را به عنوان یک نشانه در نظر گرفت که هر چه غلیظ‌تر باشد امتیاز بالاتری برای آن راه حل دارد.**

**2-2-2- عملکرد الگوریتم مورچگان**

**الگوریتم ACO از همین رفتار “به جا گذاشتن فرمون از مسیر لانه تا غذا توسط مورچه‌ها و دنبال کردن مسیر توسط سایر مورچه ها برای رسیدن به غذا” الهام گرفته شده است.**

**ویژگی های اصلی الگوریتم کلونی مورچگان عبارتند از : بازخورد مثبت[[42]](#footnote-42)، محاسبات توزیع شده[[43]](#footnote-43) و پی برنده(اکتشافی) حریصانه[[44]](#footnote-44). بازخورد مناسب برای کشف سریع راه حل های خوب موثر است، محاسبات توزیع شده مانع همگرایی زود هنگام[[45]](#footnote-45) می شوند و اکتشافی حریص به یافتن راه حل های قابل قبول در مراحل اولیه فرآیند جستجو کمک می نماید]15[.**

**2-2-3- شبه کد الگوریتم مورچگان**

1. **ایجادجمعیت اولیه**
2. **محاسبه تابع ارزیاب**
3. **تعیین مقدار فرومون**
4. **مراحل زیر تا رسیدن به شرایط خاتمه تکرار می گردد:**
5. **ساخت منابع جدید با توجه به فرومون**
6. **به روزرسانی فرومون**

**2-2-4-کاربرد الگوریتم مورچگان**

**الگوریتم‌های بهینه‌سازی مورچگان در حل مسائل بهینه‌سازی ترکیبی کاربرد بسیاری دارد، ازجمله تخصیص درجه دو، یا مسیریابی وسایل نقلیه و بسیاری از روش های به کار گرفته شده برای حل مسائل پویا[[46]](#footnote-46) با متغیرهای واقعی، مسائل تصادفی، چند هدفه و پیاده‌سازی‌های موازی. همچنین این الگوریتم‌ برای تولید پاسخ‌های شبه-بهینه برای مسئله فروشنده دوره‌گرد نیز به کار رفته‌اند. الگوریتم کلونی مورچگان می‌تواند به صورت پیوسته اجرا شده و خود را با تغییرات به صورت بی‌درنگ سازگار کند. این قضیه در مسیریابی شبکه و سیستم‌های نقل و انتقال شهری دارای اهمیت است.**

**2-3- الگوریتم زنبورها**

**این الگوریتم در سال 2005 در مقاله ای با عنوان الگوریتم زنبورها ]17[ توسط فام[[47]](#footnote-47) و همکارانش معرفی شد.الگوریتم زنبورها یک الگوریتم جمعیت محور است که براساس رفتار زنبورها در یافتن غذا می باشد.**

**فرآیند کاوش برای یافتن غذا با فرستادن زنبورهای اسکات[[48]](#footnote-48) به خارج از کلونی شروع می شود. زنبورهای اسکات به صورت تصادفی منابع غذایی را می یابند و میزان خوبی و سودمند بودنشان را ارزیابی می نمایند. زمانی که آن‌ها به کندو باز می گردند، مواد غذایی برداشت شده را ذخیره می‌کنند. آن دسته از زنبورهایی که منبع غذایی بسیار سود آوری پیدا کردند به یک منطقه در کندو به نام پیست رقص رفته و آیینی به نام رقص حرکتی[[49]](#footnote-49) را اجرا می‌کنند. طول این رقص متناسب با امتیاز اسکات از منبع غذایی است، یعنی هرچه زمان رقص یک زنبور بیشتر باشد نشان دهنده این است که منبع غذایی یافته شده توسط آن زنبور بهتر است و در نتیجه زنبورهای کاوشگر بیشتری برای برداشت به آن منبع فرستاده می‌شوند. بعد از رقص، اسکات برای جمع‌آوری بیشتر غذا به محلی که کشف کرده‌است می‌رود. تا زمانی که این محل‌ها سود آور تلقی شوند، موقع برگشت این منابع غذایی غنی توسط اسکات ها تبلیغ می‌شوند**.

**الگوریتم زنبور با الهام از نحوه عملکرد زنبورها در یافتن منابع غذایی به بهینه سازی مسائل می پردازد. این الگوریتم هر پاسخ احتمالی را به عنوان یک منبع غذایی تحت بررسی قرار می دهد. به وسیله تابع ارزیاب میزان کیفیت پاسخ احتمالی، که معادل منبع غذایی یافته شده توسط زنبور اسکات است، رامحاسبه می نماید. پاسخ ها رتبه بندی شده و جستجو در مکان های با رتبه بهتر با تمرکز بیشتری صورت می گیرد.**

**2-3-1- شبه کد الگوریتم زنبور**

1. **ایجاد جمعیت اولیه**
2. **محاسبه تابع ارزیاب برای جمعیت**
3. **انتخاب تعدادی از بهترین منابع**
4. **مراحل زیر تا رسیدن به شرایط خاتمه تکرار می گردد:**
5. **اعمال جستجوی محلی[[50]](#footnote-50) (شعاعی از بهترین منابع مشخص شده را جستجو می نماید)**
6. **اعمال جستجوی سراسری[[51]](#footnote-51)(جستجوی تصادفی)**

**2-3-2- کاربردهای الگوریتم زنبور**

**این الگوریتم در بهینه سازی سیستم های خوشه بندی[[52]](#footnote-52) و طبقه بندی، ساخت[[53]](#footnote-53)، کنترل[[54]](#footnote-54) وبهینه سازی های چند هدفه[[55]](#footnote-55) کاربردهای فراوانی دارد.**

**2-4- الگوریتم ازدحام ذرات[[56]](#footnote-56)**

**این الگوریتم ]18[ در سال 1995 توسط کندی[[57]](#footnote-57) به منظور بهینه سازی توابع غیرخطی[[58]](#footnote-58) ارائه شد. منبع الهام اين الگوريتم، رفتار اجتماعي حيوانات، همانند حرکت دسته جمعي پرندگان و ماهي‌ها می باشد. اساس کار این الگوریتم این است که در هر لحظه هر ذره مکان خود را در فضای جستجو باتوجه به بهترین مکانی که تاکنون در آن قرار گرفته است و بهترین مکانی که در کل همسایگی اش وجود دارد، تنطیم می نماید. برطبق این روش هر ذره برای آن که تغییر مناسبی را انجام دهد به اطلاعات زیر نیازمند است:**

* **بهترین عمومی : بهترین مکانی در بین تمامی ذرات**
* **بهترین همسایگی : از طریق ارتباط با ذرات دیگر از بهترین های موجود در همسایگی اش مطلع می شود**
* **بهترین محلی : بهترین مکانی که خود تا به حال در آنجا بوده**

**همه ذرات سعی می نمایند که خود را به بهترین عمومی نزدیک کنند.**

**الگوریتم ازدحام ذرات با داشتن اطلاعات فوق توسط فرمول زیر مکان ذرات را به روز رسانی می نماید:**

\*\*()

*بردار سرعت i امین ذره در تکرار* *t ام*

: *ضریب وزنی اینرسی(حرکت در مسیر خودی) که نشان دهنده تاثیر بردار سرعت در تکرار قبل برروی بردار سرعت در تکرار بعدی است.*

: *بردار سرعت i امین ذره در تکرار-1* *t ام*

: ضریب ثابت آموزش (حرکت در مسیر بهترین مقدار ذره مورد بررسی)

: دو عدد تصادفی با توزیع یکنواخت در بازه 0 تا 1

: بهترین موقعیتی که ذره تاکنون داشته

: بردار موقعیت در تکرار *(t-1)* ام

: ضریب ثابت آموزش (حرکت در مسیر بهترین ذره یافت شده در بین کل جمعیت)

: بهترین موقعیتی که بهترین ذره در بین کل ذرات تاکنون داشته

: بردار موقعیت در تکرار *t* ام

2-4-1- شبه کد الگوریتم ازدحام ذرات

1. ایجاد جمعیت اولیه
2. تا رسیدن به شرایط توقف، مراحل زیر تکرار می گردد:
3. برای هر ذره:
   1. محاسبه تابع ارزیاب
   2. اگر تابع ارزیاب این ذره، بهتر از است، مقدار آن را به عنوان جدید قرار بده
4. برای هر ذره:
   1. در همسایگی ذره به دنبال بهترین تابع ارزیاب بگرد
   2. سرعت ذره را محاسبه کن
   3. موقعیت ذره را به روز رسانی کن

2-4-2- کاربردهای الگوریتم ازدحام ذرات

الگوریتم ازدحام ذرات یک روش بهینه سازی سراسری است که با استفاده از آن می‌توان مسائلی که جواب آنها یک نقطه یا سطح در فضای n بعدی می‌باشد، را بهینه نمود.

**2-5- الگوریتم تبرید شبیه سازی شده**

**الگوریتم تبرید]22[ توسط کریک پاتریک [[59]](#footnote-59)و همکارانش در سال 1983 ارائه شد. این الگوریتم برای بهینه سازی در مسائل با فضای جستجوی بزرگ موثر است. در این پژوهش فرآیند سرد سازی تدریجی[[60]](#footnote-60) برای بهبودخواص کیفی کریستال جامد، به منظور حل مسائل بهینه سازی ترکیباتی شبیه سازی شده است. ایده اصلی این الگوریتم این است که ممکن است گاهی یک حرکت یا جواب بد مقدمه ای برای رسیدن به یک جواب خوب باشد بنابراین به جای آن که همیشه جواب های بد را دور ریخته و فقط به سوی پاسخ های خوب برویم، پاسخ های بد را، با درصد کمتری نسبت به پاسخ های خوب، مورد بررسی قرار دهیم.**

**در الگوریتم شبیه سازی تبرید هر نقطه یا پاسخ احتمالی در فضای جستجو معادل حالتی از یک سیستم فیزیکی است و تابع هزینه مساله که باید کمینه شود معادل با انرژی داخلی سیستم در آن حالت است و هدف رسیدن به حالتی است که سیستم کمترین انرژی را داشته باشد. الگوریتم در هر مرحله یک حالت را در نظر گرفته و به صورت احتمالی میان ماندن در حالت فعلی یا انتقال به حالت جدید تصمیم گیری می نماید. این عمل تا زمانی که سیستم به حالتی با انرژی مطلوب برسد و یا اینکه میزان محاسبات از آستانه[[61]](#footnote-61) تعیین شده بیشتر شود.**

**2-5-1- شبه کد الگوریتم تبرید شبیه سازی شده**

1. **ایجاد جمعیت اولیه**
2. **محاسبه تابع ارزیاب**
3. **تا رسیدن به شرط توقف مراحل زیر تکرار می شود:**
4. **یک نقطه را در همسایگی انتخاب کن**
5. **تابع ارزیاب ان را محاسبه کن**
6. **اگر اختلاف نقطه با همسایه کمتر از صفر بود، نقطه همسایه را بپذیر**
7. **در غیر این صورت، به آن احتمال پذیرش اختصاص بده**

**2-5-2- محاسبه احتمال پذیرش[[62]](#footnote-62)**

**احتمال پذیرش عبارتست از احتمال تغییر حالت کنونی به یک حالت کاندید. اگر حالت کنونی را و حالت کاندید را و دمای متغیر با زمان سیستم را با نشان دهیم. احتمال پذیرش برابر اس با :**

**2-5-3-کاربردهای الگوریتم تبرید شبیه سازی شده**

**الگوریتم تبرید شبیه سازی شده در مسائل بهینه سازی بسیاری مورد استفاده قرار می گیرد از جمله مسائل زمانبندی فلوشاپ، طرح زمان بندی داروی سرطان، برای طرح ریزی شیمی درمانی های با چرخه خاص[[63]](#footnote-63).**

**2-6- الگوریتم بهینه سازی رقابت استعماری[[64]](#footnote-64)**

**الگوریتم رقابت استعماری ]52[ در سال2007 توسط اسماعیل آتش پز ارائه شد که از یک پدیده اجتماعی – انسانی الهام گرفته است. بطور ویژه این الگوریتم به فرایند استعمار، به عنوان مرحله‌ای از تکامل اجتماعی – سیاسی بشر نگریسته و با مدل‌سازی ریاضی این پدیده، از آن به عنوان منشأ الهام یک الگوریتم قدرتمند در زمینه بهینه‌سازی بهره می‌گیرد.**

**همانند الگوریتم های بهینه سازی تکاملی دیگر، الگوریتم راقبت استعماری هم با یک جمعیت اولیه شروع به کار می نماید. در این روش ب هر پاسخ احتمالی کشور[[65]](#footnote-65) نامیده می شود. که به دو صورت استعمارگر ها و مستعمره ها[[66]](#footnote-66) می باشند و این دو باهم یک فرمانروایی[[67]](#footnote-67) را ایجاد می نمایند.** **تعدادی از بهترین عناصر جمعیت به عنوان استعمارگر انتخاب می‌شوند. باقی‌مانده جمعیت نیز به عنوان مستعمره، در نظر گرفته می‌شوند. اساس کار این الگوریتم، رقابت مستعمره ها[[68]](#footnote-68) می باشد. در طول این رقابت فرمانروایی های ضعیف تر سقوط کرده و قوی ترها مستعمرات آن ها را تصاحب می نمایند. این رقابت به سمتی پیش می رود که در انتها تنها یک فرمانروایی باقی بماند.**

**2-6-1- حرکت مستعمره ها به سمت استعمارگرها**

**همانطور که در زندگی انسان ها زمانی که کشوری به استثمار درمی آید، کشور استعمارگر هم از نظر فرهنگی و هم از نظر اجتماعی بر کشور مستعمره تاثیر می گذارد. به عنوان مثال کشورهایی نظیر انگلیس و فرانسه با تعقیب سیاست همگون‌سازی در مستعمرات خود در فکر ایجاد انگیس نو و فرانسه نو در مستعمرات خویش بودند. این بخش از فرایند استعمار در الگوریتم بهینه‌سازی رقابت استعماری، به صورت حرکت مستعمرات به سمت کشور استعمارگر، مدل شده‌است]53[.**

**به منظور پیاده سازی این فاز از الگوریتم، کشور مستعمره، به اندازه x واحد در جهت کشوراستعمارگر حرکت می نماید و به موقعیت جدیدکشانده می‌شود. x عددی تصادفی است که می توان آن را با توزیع یکنواخت (و یا هر توزیع مناسب دیگر) بدست آورد.**

**2-6-2-انقلاب[[69]](#footnote-69)**

**انقلاب تغییراتی ناگهانی را در ویژگی‌های اجتماعی - سیاسی یک کشور ایجاد می‌کند. در الگوریتم رقابت استعماری، انقلاب با جابجایی تصادفی یک کشور مستعمره به یک موقعیت تصادفی جدید مدلسازی می‌شود.**

**2-6-3- جابجایی مستعمره و استعمارگر**

**در طول تغییرات و حرکت مستعره ها به سوی استعمارگرها، ممکن است یک مستعمره به موقعیت بهتری نسبت به کشور استعمارگر برسد. در این صورت این دو با یکدیگر جایگزین می شوند و کشور مستعمره، حال کشور استعمارگر می شود.**

**2-6-4- شبه کد الگوریتم رقابت استعماری**

1. **ایجاد جمعیت اولیه**
2. **حرکت مستعمرات به سمت کشور استعمارگر**
3. **عملگر انقلاب**
4. **درصورتی که مستعمره ای بهتر از استعمارگر بود، جای آنها را عوض کن**
5. **محاسبه هزینه کل یک استعمارگر**
6. **یک (یا چند) مستعمره از ضعیف‌ترین فرمانروایی انتخاب کرده و آن را به فرمانروایی که بیشترین احتمال تصاحب را دارد، بده.**
7. **حذف فرمانروایی های ضعیف**
8. **اگر تنها یک فرمانروایی باقی‌مانده باشد، توقف کن در غیر این صورت به گام ۲ برو**

**2-6-5- کاربردهای الگوریتم رقابت استعماری**

**در حالت کلی الگوریتم رقابت استعماری به هر نوع مسئله بهینه‌سازی بدون هیچ محدودیتی قابل اعمال است. همچنین این الگوریتم در طراحی بهینه کنترل کننده برای سیستمهای صنعتی همچون سیستم چند ورودی - چند خروجی تبرید، صنعتی ستون تقطیر ، طراحی بهینه سیستم‌های پیشنهاد دهی هوشمند ، طراحی بهینه آنتن‌های آرایه‌ای، حل مسائل** [**برنامه‌ریزی تولید**](https://fa.wikipedia.org/wiki/%D8%A8%D8%B1%D9%86%D8%A7%D9%85%D9%87%E2%80%8C%D8%B1%DB%8C%D8%B2%DB%8C_%D8%AA%D9%88%D9%84%DB%8C%D8%AF) **در حوزه** [**مهندسی صنایع**](https://fa.wikipedia.org/wiki/%D9%85%D9%87%D9%86%D8%AF%D8%B3%DB%8C_%D8%B5%D9%86%D8%A7%DB%8C%D8%B9) **و مدیریت ،یادگیری و تحلیل** [**شبکه‌های عصبی**](https://fa.wikipedia.org/wiki/%D8%B4%D8%A8%DA%A9%D9%87%E2%80%8C%D9%87%D8%A7%DB%8C_%D8%B9%D8%B5%D8%A8%DB%8C) **مصنوعی، طراحی بهینه موتورهای القای خطی مورد استفاده قرارا می گیرد.**

**2-7- الگوریتم کرم شب تاب[[70]](#footnote-70)**

**الگوریتم کرم شب تاب به اختصار FA در اواخر سال ۲۰۰۷ و توسط زین شی یانگ[[71]](#footnote-71) معرفی شده است، که ایده اصلی آن از ارتباط نوری میان کرم های شب تاب الهام گرفته شده است. این الگوریتم را می توان از مظاهر هوش ازدحامی دانست، که در آن از همکاری (و احتمالا رقابت) اعضای ساده و کم هوش، مرتبه بالاتری از هوشمندی ایجاد می شود که قطعا توسط هیچ یک از اجزا قابل حصول نیست.**

**بطور کلی سه قانون ایده آل براي توسعه الگوریتم هاي الهام گرفته شده از کرم شب تاب بکار برده شده است که عبارتند از]54[:**

* **همه کرم هاي شب تاب از یک جنس در نظر گرفته می شوند ، بگونه اي که یک کرم شب تاب به کرم شب تاب دیگر صرفنظر از جنسیت شان جذب خواهد شد.**
* **جذابیت با روشنایی یک کرم شب تاب متناسب است. بنابراین براي هر کرم شب تاب چشمک زن ، یکی با روشنایی کمتر به سوي دیگري با درخشندگی بیشتر حرکت خواهد کرد. جذابیت با روشنایی متناسب است و هردوي آن ها هرچه که فاصله افزایش می یابد،کاهش می یابند. اگر هیچ یک از کرم هاي شب تاب روشنتر از دیگري نباشد ، کرم شب تا ب بطور تصادفی حرکت خواهد کرد.**
* **روشنایی یک کرم شب تاب بوسیله چشم انداز تابع هدف تعیین می شود یا تحت تاثیر آن قرار می گیرد.**

**الگوریتم کرم شب تاب، یک الگوریتم جمعیت محور است. نور چشمک زن و جذابیت حاصل از آن را می توان به طرق مختلفی فرموله سازی نمود.**

**این الگوریتم با قرار دادن جمعیتی n عضوی از کرم های شب تاب در نقاط مختلف فضای جستجوی مسئله بهینه سازی به صورت تصادفی آغاز می شود. در ابتدا همه ی کرم ها مقدار یکسانی از لوسی فرین به اندازه ی یک در اختیار دارند. هر تکرار الگوریتم شامل یک فاز به روز کردن لوسی فرین و یک فاز به روز کردن مکان کرم ها می باشد:**

**مقدار لوسی فرین هر کرم در هر تکرار با توجه به مقدار برازندگی موقعیت آن کرم تعیین می شود و در هر تکرار با توجه به مقدار برازندگی و متناسب با آن مقداری به لوسی فرین[[72]](#footnote-72) فعلی کرم افزوده می شود.**

**در خلال فاز حرکت، هر کرم به صورت احتمالاتی به سمت یکی از همسایگانش که لوسی فرین بالاتری دارد حرکت می کند. به این ترتیب کرم ها به سمت همسایگان با درخشندگی بیشتر حرکت می کنند.**

**2-7-1- شبه کد الگوریتم کرم شب تاب**

1. **ایجاد جمعیت اولیه**
2. **محاسبه درخشندگی جمعیت براساس تابع ارزیاب**
3. **تا رسیدن به شرایط توقف مراحل زیر تکرار می شود:**
4. **تمامی کرم ها را نسبت به هم ارزیابی نموده و درصورتی که درخشندگی کرمی بیش از دیگری بود، کرم با درخشندگی کمتر را به سمت کرم با درخشندگی بیشتر حرکت بده**
5. **درخشندگی کرم های جدید را محاسبه کن**

**2-7-2- کاربرد الگوریتم کرم شب تاب**

**الگوریتم کرم شب تاب در سال های اخیر توجه بسیاری از محققان را به خود جلب نموده است. از جمله کاربردهای این الگوریتم می توان به موارد زیر اشاره نمود ]54[:**

**حداقل سازي زمان محاسباتی براي تراکم تصویر دیجیتال ، بررسی ثبت تصویر، مساله انتخاب ویژگی، مسائل طراحی چند کیفیتی غیر خطی، حل مسائل بارگذاري اقتصادي با کاهش دادن اتلاف توان.**

**2-8- الگوریتم بهینه سازی فاخته**

**الگوریتم فاخته ]1[ در سال 2011 توسط رامین رجبیون[[73]](#footnote-73) معرفی گردید. این الگوریتم از سبک زندگی و نحوه تخم گذاری و پرورش تخم های پرنده ای با نام کوکو یا فاخته الهام گرفته شده است.** **فاخته** [**تخم**](https://fa.wikipedia.org/wiki/%D8%AA%D8%AE%D9%85_(%D8%B2%DB%8C%D8%B3%D8%AA%E2%80%8C%D8%B4%D9%86%D8%A7%D8%B3%DB%8C)) **خود را در آشیانه پرندگان دیگر می‌گذارد تا پرنده صاحب لانه، جوجه را بزرگ کند. کاری که** [**جوجه‌گذاری انگلی**](https://fa.wikipedia.org/w/index.php?title=%D8%AC%D9%88%D8%AC%D9%87%E2%80%8C%DA%AF%D8%B0%D8%A7%D8%B1%DB%8C_%D8%A7%D9%86%DA%AF%D9%84%DB%8C&action=edit&redlink=1) **نامیده می‌شود. گاه پرنده میزبان متوجه حضور تخم بیگانه در آشیانه خود شده و یا آن را دور می اندازد و یا آشیانه خود را ترک می کند و به جای دیگری می رود و گاه نیز متوجه این موضوع نمی شود و تخم فاخته امکان رشد و بزرگ شدن را در آن لانه به دست می آورد. الگوریتم جستجوی فاخته نیز براساس چنین شیوه ای عمل می نماید. ابتدا مانند تمامی الگوریتم های تکاملی با یک جمعیت اولیه تصادفی شروع می نماید و در طول فرآیند با اختصاص دادن تعداد تصادفی تخم به هر فاخته و تعیین شعاع تخم گذاری برطبق فرمول خاص جستجو را انجام می دهد. شعاع تخم گذاری فاخته ها توسط فرمول زیر محاسبه می شود:**

**که در این فرمول یک عدد صحیح است.**

**2-8-1- شبه کد الگوریتم فاخته**

1. **ایجاد جمعیت اولیه**
2. **تا رسیدن به شرایط خاتمه مراحل زیر تکرار می شود:**
3. **اختصاص تعدادی تخم به هر فاخته**
4. **محاسبه شعاع تخم گذاری برای هر فاخته**
5. **تخم گذاری فاخته ها با توجه شعاع مجاز**
6. **از بین بردن تخم های نامطلوب**
7. **رشد تخم های باقی مانده**
8. **ارزیابی تخم ها**
9. **طبقه بندی فاخته ها و یافتن بهترین گروه و انتخاب موقعیت هدف**
10. **رفتن فاخته ها به سمت موقعیت هدف**

**فصل سوم**

**توضیح روش پیشنهادی**

**در این فصل به توضیح سبک زندگی کوآلاها و نحوه تغذیه شان پرداخته شده است. سپس نحوه بهینه سازی الگوریتم الهام گرفته شده از این جاندار را ارائه می شود.**

**3-1- سبک زندگی کوآلاها**

**خرس کوآلا جانوری گیاهخوار و کیسه‌دار و بومی استرالیا است.کوآلاها فقط در جنگل‌های اُکالیپتوس[[74]](#footnote-74) استرالیا زندگی می‌کنند. طول این حیوان بین 69 تا 79 سانتیمتر (27 تا 31 اینچ) و وزنش بین 5 الی 12 کیلوگرم (11 تا 26 پوند ) مى باشد. معمولأ جنس نر این حیوان، بزرگتر ازنوع ماده مى باشد و کوالاهایى که درقسمت جنوبى زندگى مى کنند از انواع شمالى نسبتأ بزرگترهستند**.**آنها هر شب با دستهای پر قدرتشان شاخه‌ها را می‌گیرند و شاخه به شاخه برگهای سخت اکالیپتوس را می‌خورند. میانگین وزن برگ‌های اکالیپتوس که می‌خورند ۳۰۰ الی ۴۰۰ گرم است. این برگها ارزش غذایى پایینى دارند زیرا شامل مقدار ناچیزى پروتئین بوده و قسمت زیادى ازآن را فیبر ودیگر ترکیبات غیرقابل هضم تشکیل داده است. کوالا به طور منحصر به فردى، با این نوع تغذیه سازگارى پیدا کرده است و این سازگارى، فرایند گوارش حیوان را از بسیارى جهات همچون شکل دندانها و سرعت هضم غذا، تحت تاثیر قرار داده است.** **خرسهای کوآلا به علت خوردن برگهای اکالیپتوس همیشه بوی همین برگ را می‌دهند]24[. آنها روزها بر روی شاخه‌های پهن اکالیپتوس می‌خوابند و بچه شان را هم آنجا به دنیا می‌آورند.**

**کوالا براى جبران مادۀ غذایى کم انرژى وغیر مغذى خود بیشتر وقت خود را بر روى درختان به استراحت مى پردازد تا بتواند انرژى خود را ذخیره کند. آنها 20 ساعت در روز مى خوابند، 1 تا 3 ساعت را به نشستن و خوردن غذا اختصاص مى دهند و تنها 4 دقیقه در روز فعالانه بین شاخه هاى درختان در حرکتند و به تمیز کردن کرکهاى خود و شرکت در فعالیتهاى اجتماعى مى پردازند. آنها عمدتأ از ساعت 5 عصر تا نیمه شب تغذیه مى کنند. کوالاها در چندین نوبت غذا مى خورند و هر وعدۀ غذایى به طور متوسط بیست دقیقه طول مى کشد.**

**کوالاهاى ماده در ۱۸ الى ۲۴ ماهگى ازنظر جنسى بالغ مى شوند. کوالاهاى بزرگتر و نیرومندتر، قادرند در سن پایین تولید مثل کنند. با این حال، یک کوالاى نر باید براى رقابت با نرهاى دیگردر پیدا کردن جفت، به اندازۀ کافى بزرگ شده باشد وبه طورکلى سن جفت گیرى این حیوان حدود ۴ سالگى است.**

**وقتی کوالاها متولد می شوند فقط 2 سانتی متر طول دارند نابینا بوده و گوشهای آنها هنوز توسعه پیدا نکرده اند . پس از رفتن به داخل کیسه آنها خود را به یکی از دو پستانی که داخل کیسه وجود دارد چسبانده ، پستان در داخل دهان نوزاد متورم شده و باعث می شود که آنها از این منبع غذایی جدا نشده و از جای خود خارج نشوند . نوزاد برای مدت 6 یا 7 ماه در کیسه ی مادرش مانده و در طول این مدت فقط شیر می نوشد. پس از خارج شدن از کیسه ، نوجوان در روی شکم یا بر پشت مادرش حمل می شود . اگر چه بازگشت به کیسه ی مادر به منظور تغذیه از شیر تا زمانی که کوالا به اندازه ای بزرگ بشود که در داخل کیسه جا نشود ادامه می یابد.**

**کوآلاهای والد در سن ۶ تا ۷ ماهگی فرزندان خود ماده‌ای را به عنوان غذا به آنها می‌دهند که در اصل نوع ویژه‌ای از** [**مدفوعشان**](https://fa.wikipedia.org/wiki/%D9%85%D8%AF%D9%81%D9%88%D8%B9) **است. باکتری‌هایی در دستگاه گوارشی این** [**پستانداران**](https://fa.wikipedia.org/wiki/%D9%BE%D8%B3%D8%AA%D8%A7%D9%86%D8%AF%D8%A7%D8%B1%D8%A7%D9%86) **وجود دارند که به آن‌ها کمک می‌کنند رشته‌های سلولزی موجود در برگ اکالیپتوس را بشکنند. در هنگام گذار از شیر مادر به تغذیه اکالیپتوسی، مادر نخست با کمک آن مدفوع به فرزند خود میکروارگانیسم‌های مورد نیاز برای هضم گیاه را می‌خوراند. پس از آنکه بدن فرزند پروتئین‌های مورد نیاز برای کمک به هضم را به دست‌آورد، تغذیه اصلی با برگ اکالیپتوس آغاز می‌شود ]25[.**

**این مساله که در این جانداران والد پس از تولید مثل شروع به تغذیه خود می نماید و تا زمانی که بچه کوآلا بتواند خود به دنبال غذا رفته و معده اش آماده هضم اکالیپتوس شود، از والد تغذیه می شود ایده اصلی الگوریتم پیشنهادی است.**

**3-2- روش پیشنهادی**

**ایجاد جمعیت اولیه به صورت تصادفی**

**ترکیب**

**تغذیه والد**

**تغذیه فرزند توسط والد**

**جایگرین نمودن جمعیت جدید**

**آیا شرط توقف ارضا شده است**

**خیر**

**بله**

**پایان**

**شکل 3-1 فلوچارت الگوریتم پیشنهادی**

**3-3- شرح الگوریتم پیشنهادی**

**شکل 3-1 نشان دهنده فلوچارت الگوریتم کوآلا می باشد. همانطور که مشاهده می شود، الگوریتم با ساخت یک جمعیت اولیه[[75]](#footnote-75) تصادفی شروع می نماید. در واقع این جمعیت اولیه نماینده کوآلاهای بالغ می باشند. هریک از این کوآلا ها برحسب میزان قدرت و بالغ بودن شروع به جفت گیری می نمایند. هرچه کوآلا قوی تر باشد احتمال جفت گیری بیشتری دارد. پس از تولید مثل، والدین به تغذیه[[76]](#footnote-76) می پردازند تا انرژی مصرف شده را به خود باز گردانند. در این حین تا زمانی که نوزاد کوآلا به بلوغ برسد و آمادگی زندگی مستقل را داشته باشد، توسط والدین(ابتدا از طریق شیر مادر و سپس با مدفوع والدین) تغذیه می شود و از نظر جسمی بهبود می یابد و رشد می نماید.**

**3-3-1- ایجاد جمعیت اولیه**

**یک مبحث مهم در حل مسائل بهینه سازی، این است که متغیرهای مساله به درستی در یک آرایه گنجانده شوند]1[. این آرایه در الگوریتم ژنتیک با عنوان کروموزوم[[77]](#footnote-77)، در الگوریتم زنبور با نام زنبور اسکات، در الگوریتم ازدحام ذرات با عنوان موقعیت ذره[[78]](#footnote-78) و در الگوریتم فاخته با عنوان زیستگاه[[79]](#footnote-79) بیان می شوند. در الگوریتم پیشنهادی این آرایه، کوآلا نامیده می شود. اگر مساله بهینه سازی مدنظر N متغیره باشد آنگاه آرایه کوآلای بیان شده، یک آرایه بعدی است که به صورت زیر است:**

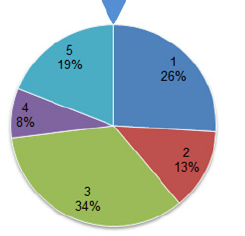
**ارزیابی کوآلاها را توسط تابع سودمندی[[80]](#footnote-80) محاسبه می نماییم.**

**برای شروع الگوریتم، یک ماتریس بعدی متشکل از کوآلاهایی که به صورت تصادفی مقداردهی شده اند ایجاد می شود.**

**3-3-2- انتخاب[[81]](#footnote-81)**

**این عملگر در بیشتر الگوریتم های بهینه سازی متاهیوریستیک وجود دارد و روش های مختلفی برای آن ارائه شده است از جمله چرخ رولت[[82]](#footnote-82)، روش انتخاب نخبگان[[83]](#footnote-83)، روش مقیاس گیری[[84]](#footnote-84) و روش تورنمنت[[85]](#footnote-85).**

**روش به کار برده شده در این پژوهش، روش انتخاب چرخ رولت است. چرخ رولت یا انتخاب متناسب با سودمندی[[86]](#footnote-86) عملگری است که برای انتخاب راه حل هایی با پتانسیل سودمندی بیشتر برای ترکیب انتخاب می شوند.**



**شکل 3-2 چرخ رولت**

**در چرخ رولت ابتدا براساس تابع ارزیاب، میزان سودمندی هر یک از راه حل ها سنجیده و سپس براساس این مقدار، یک احتمال به هریک از راه حل ها انتساب داده می شود. هرچه میزان سودمندی راه حل بیشتر باشد احتمال انتساب داده شده به آن نیز بیشتر شده و درنتیجه شانس آن برای انتخاب شدن به منظور ترکیب بیشتر می شود. روش های مختلفی برای محاسبه احتمالات مذکور موجود است. در ساده ترین حالت برای بدست آوردن احتمال i-امین عنصر می توان از فرمول زیر استفاده نمود]31[:**

**که مقدار تابع ارزیاب برای عنصر i-ام می باشد.**

**3-3-3- عملگر ترکیب[[87]](#footnote-87)**

**عملگر ترکیب برای تولید نسل جدید است که از نحوه ی بازتولید[[88]](#footnote-88) و ترکیب بیولوژیکی در طبیعت الهام گرفته شده است. در واقع در این فرآیند بیش از یک والد انتخاب شده و ترکیب می شوند و نسل فرزندان را به وجود می آورند. روش های متفاوتی برای انجام این عملگر وجود دارد که در بخش 2 به توضیح برخی از آنها پرداخته شد.**

**در این پژوهش برای ساخت فرزندان از ترکیب کوآلاهای والد از روش ترکیب یکنواخت استفاده شده است.**

***For each parent;***

***Create a random binary array in size of parent***

***Child1: parent1+ (1- ) parent2***

***Child2: parent2+ (1- ) parent1***

**3-3-4- عملگر تغذیه والد[[89]](#footnote-89)**

**پس از تولید مثل، کوآلاها مدتی را به تغذیه و بهبود سطح انرژی بدنشان می پردازند.در این پژوهش با الهام از این عمل کوآلاها، عملگری با عنوان تغذیه والد ارائه شده است. الگوریتم پس از آن که توسط چرخ رولت والدین را انتخاب و عمل ترکیب را اعمال نمود، سعی در بهبود منابع والد می نماید. برای شبیه سازی عمل کوآلاها، برای یک تعداد تکرار معین برای هر عنصر، بخشی را به تصادف تغییر می دهیم. در بین تغییرات ایجاد شده،موقیعت تغییری که سبب سودمندی بیشتر و ارتقا تابع ارزیاب مربوطه می شود را با والد اولیه جایگزین می نماییم.**

**{**

**{**

**}**

**}**

**هرچه نرخ تغییرات اعمالی بیشتر باشد، جستجوی سراسری بیشتری انجام می شود و در صورتی که نرخ آن کم باشد جستجوی محلی صورت می گیرد،**

**3-3-5- عملگر تغذیه فرزندان[[90]](#footnote-90)**

**پس از آن که والد انرژی از دست رفته را بدست آورد فرزند خود را تغذیه می نماید و سعی می نماید که آن را بهبود دهد. برای نیل به این هدف و شبیه سازی این عمل، در الگوریتم پیشنهادی عملگری با عنوان تغذیه فرزندان را ارائه نمودیم که سعی می کند با درصدی اختلاف به سمت والد خود برود. در صورتی که در این مسیر بهبود یافت و سودمندی آن افزایش یافت فرزند جدید را جایگزین می نماییم در غیر این صورت بدون تغییر می ماند.**

**{**

**}**

**3-3-6- جستجوی اکتشافی[[91]](#footnote-91) و استخراجی[[92]](#footnote-92)**

**اکتشاف و استخراج دو مفهوم مهم در مسائلی است که به روش جستجو حل می شوند]36[. اکتشاف فرآیندی است که طی آن مناطق و نواحی جدید از فضای جستجو بررسی و کشف می گردد در حالیکه استخراج، جستجوی بیشتر و با تمرکز بیشتر در مناطق قبلا دیده شده است. هر الگوریتم جستجو نیاز دارد که این دو نوع جستجو را در خود داشته باشد. در الگوریتم پیشنهادی کوآلا دو عملگر نوین ارائه شده است که عبارتند از تغذیه والد و تغذیه فرزند. در عملگر تغذیه والد هرچه نرخ تغییرات را افزایش دهیم میزان اکتشافی بودن آن افزایش می یابد که این پارامتر را می توان با توجه به نوع مساله و نیاز آن تنظیم نمود. همچنین توسط عملگر تغذیه فرزند عمل استخراج صورت می گیرد و این به علت جستجو و حرکت در اطراف و در راستای والد خود است که سبب می شود که تمرکز بیشتری در ناحیه والد شود.**

**3-3-7- ایجاد نسل جدید**

**در انتها نحوه انتخاب و جایگذاری نسل جدید می باشد. روش های مختلفی برای این کار موجود است.در برخی مسائل نسل جدید تولید شده به طور کامل با نسل اولیه جایگزین شده و در واقع نسل اولیه کاملا از بین می رود. در برخی دیگر تمامی جمعیت اولیه و جمعیت جدید تولید شده باهم ترکیب شده و مرتب می شوند و سپس به تعداد جمعیت مجاز از بهترین ها انتخاب می شود. گزینش روشی برای انتخاب جمعیت برای زنده ماندن تا حد زیادی بستگی به نوع مساله بهینه سازی و تصمیم طراح دارد.**

**فصل چهارم**

**شبیه سازی روش پیشنهادی بر روی توابع معیار**

**و مقایسه با الگوریتم های دیگر**

**در این فصل به پیاده سازی الگوریتم پیشنهادی پرداخته می شود و عملکرد آن با الگوریتم ها فاخته و ژنتیک مقایسه می گردد.**

**4-1- ارزیابی عملکرد الگوریتم های بهینه سازی**

**الگوریتم های بهینه سازی تکاملی روش های جستجوی تصادفی می باشند که براساس تکامل بیولوژیکی و یا رفتار اجتماعی گونه های مختلف عمل می نمایند. هدف از طراحی این الگوریتم های رسیدن به پاسخی بهینه یا نزدیک به بهینه در مسائلی است که به راحتی و با روش های کلاسیک موجود نمی توان آن ها را حل نمود]55[.**

**عملکرد و کارآیی الگوریتم های بهینه سازی از دو جهت سرعت همگرایی و دقت مورد بررسی قرار می گیرد. به منظور آن که عملکرد الگوریتم ها با هم مقایسه گردد، از توابع معیار یکسانی برای آنها استفاده می شود و دقت و روند همگرایی به پاسخ بهینه در آنها مورد مطالعه قرار می گیرد.**

**4-2- توابع معیار**

**در میان الگوریتم های بهینه سازی تکاملی، الگوریتم ژنتیک از محبوبترین آنهاست. اما پس از ارائه الگوریتمی با عنوان فاخته توسط رامین رجبیون توجه بسیاری به این الگوریتم جلب شد. نتایج این الگوریتم حاکی از عملکرد بسیار بهتر آن نسبت به الگوریتم ژنتیک بود. لذا در این پژوهش برای نمایش عملکرد الگوریتم پیشنهادی، الگوریتم فاخته و مقاله مربوط به آن اساس آزمایشات قرار گرفت. در مقاله فاخته پنج تابع به عنوان تابع معیار معرفی شده است که عبارتند از :**

**فرمول 4-1- تابع معیار**

**فرمول 4-2- تابع معیار**

**فرمول 4-3- تابع معیار**

**فرمول 4-4- تابع معیار**

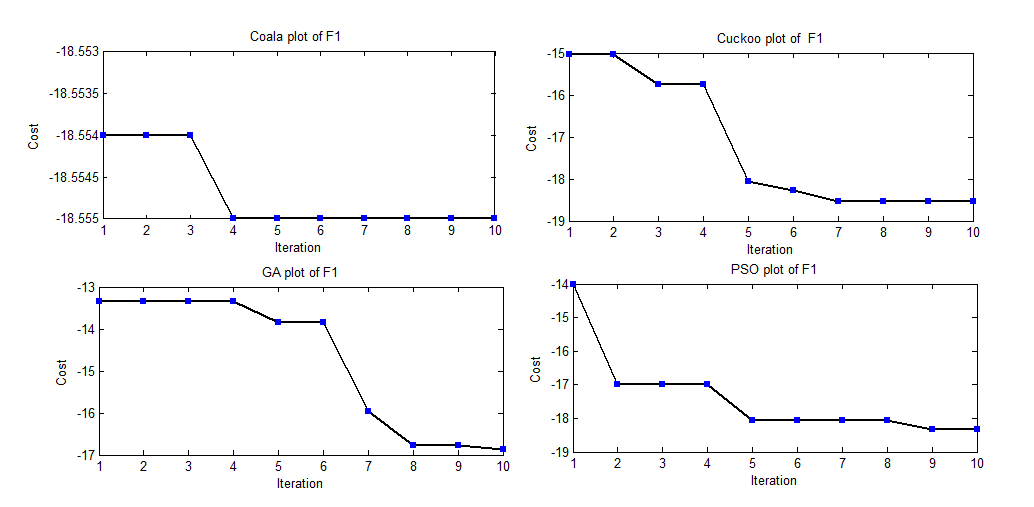
**فرمول 4-5- تابع معیار**

**4-3- پیاده سازی الگوریتم بر روی تابع معیار اول و مقایسه آن با الگوریتم های دیگر**

**در این بخش به تفکیک الگوریتم های فاخته، ژنتیک و ازدحام ذرات بر تابع معیار F1 اعمال می شود. سپس الگوریتم کوآلا پیاده سازی می شود و نتایج حاصل از پیاده سازی های صورت گرفته با یکدیگر مقایسه می گردد. در پیاده سازی های صورت گرفته تعداد جمعیت اولیه 20 و حداکثر تعداد تکرار برابر 10 در نظر گرفته شده است.**

**4-3-1- نتایج اعمال الگوریتم ها بر تابع معیار اول**

**با اعمال الگوریتم های فاخته، ژنتیک، ازدحام ذرات و الگوریتم پیشنهاد شده ی کوآلا بر اولین تابع معیار و با تنظیمات بیان شده در بخش قبل، نمودار زیر حاصل آمد.**



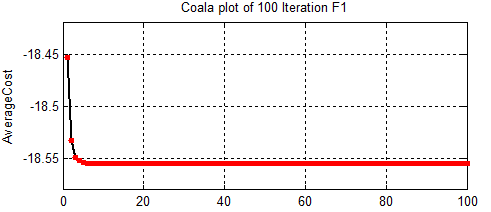
**شکل 4-6 نتایج پیاده سازی الگوریتم ها بر تابع F1**

**همانطور که می توان از شکل4-6 پی برد، در یک اجرا الگوریتم فاخته در تکرار 5 به پاسخ 18- و در تکرار 7 به 18.5- دست می یابد. الگوریتم ژنتیک در آخرین تکرار خود یعنی تکرار دهم نیز به 18- دسترسی پیدا نکرد. همچنین الگوریتم ازدحام ذرات در تکرار 5 به 18- و در تکرار 9 به 18.33- دست یافت. این در حالی است که الگوریتم کوآلا که در این پژوهش ارائه گردیده است در همان تکرار اول خود به 18.55- و در تکرار 4 به 18.555- دست یافت.**

**برای اثبات تصادفی نبودن نتایج، الگوریتم ها در 30 تکرار اجرا گردید و نتایج به شرح زیر به دست آمد. در این پیاده سازی تعداد تکرارهای داخلی هربار اجرای الگوریتم ها 100 در نظر گرفته شده است.**

**نتایج اعمال الگوریتم فاخته بر تابع اول حاکی از آن است که این الگوریتم به طور متوسط در تکرار 6.8 به بهترین پاسخ خود یعنی 18.5543- می رسد. الگوریتم ژنتیک به طور متوسط در تکرار 45.9 به پاسخ 18.5525- و الگوریتم ازدحام ذرات در تکرار 38.7 به پاسخ 18.5547 دست می یاند. اما الگوریتم کوآلا به طور متوسط در تکرار 5.26 به پاسخ 18.5545- رسیده و به طور متوسط تقریبا در تکرار5.5 به نقطه بهینه 18.5547- دست می یابد.**

**به منظور اثبات تصادفی نبودن نتایج به دست آمده الگوریتم های فاخته و کوآلا را با همان تنظیمات پارامترها، 30 بار اجرا نموده و میانگین تابع ارزیاب در این تکرار ها محاسبه گردید که نتایج را می توان در شکل زیر مشاهده نمود.**

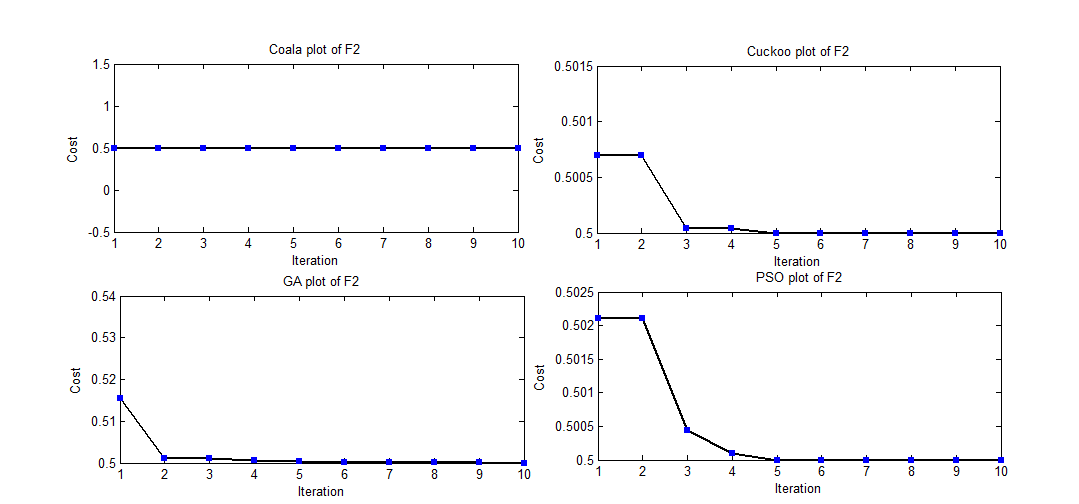
  **شکل 4-7- میانگین پاسخ های بدست آمده الگوریتم کوآلا بر F1 در 30 تکرار**

**4-4- پیاده سازی الگوریتم بر روی تابع معیار دوم و مقایسه آن با الگوریتم های دیگر**

**در این بخش الگوریتم های بیان شده به تابع معیار دوم اعمال می گردند. تعداد جمعیت اولیه در این بخش برای تمامی الگوریتم ها 20 در نظر گرفته شده است و تعداد تکرارهای داخلی در یک بار اجرای الگوریتم 10 و در 30 بار اجرای الگوریتم 100 می باشد.**

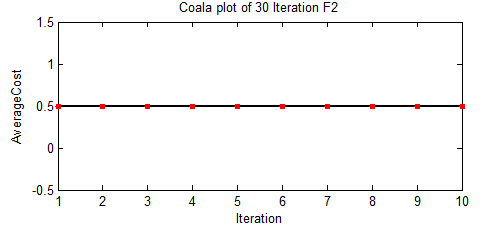
**4-4-1- نتایج اعمال الگوریتم ها بر تابع معیار دوم**

**همانطور که می توان از شکل4-8 پی برد، در یک اجرا الگوریتم فاخته در تکرار 5 به پاسخ 0.5 دست می یابد. الگوریتم ژنتیک نیز در تکرار 6 به این پاسخ دسترسی پیدا می نماید. همچنین الگوریتم ازدحام ذرات در تکرار 9 به نقطه بهینه 0.5 دست یافت. این در حالی است که الگوریتم کوآلا که در این پژوهش ارائه گردیده است در همان تکرار اول خود به پاسخ 0.5 دست یافت.**

****

**شکل 4-8 نتایج پیاده سازی الگوریتم ها روی تابع F2**

**نتایج شبیه سازی الگوریتم فاخته در 30 تکرار بر تابع معیار دوم نشان دهنده آن است که این الگوریتم به طور متوسط در تکرار 5.2 به بهترین پاسخ خود یعنی 0.5 می رسد. الگوریتم ژنتیک به طور متوسط در تکرار 12.6 و الگوریتم ازدحام ذرات در تکرار 10.3 به پاسخ دست می یباند. این در حالی است که الگوریتم کوآلا به طور متوسط در همان تکرار اول به پاسخ می رسد.**

** شکل 4-9- میانگین پاسخ های بدست آمده از الگوریتم کوآلا بر F2 در 30 تکرار**

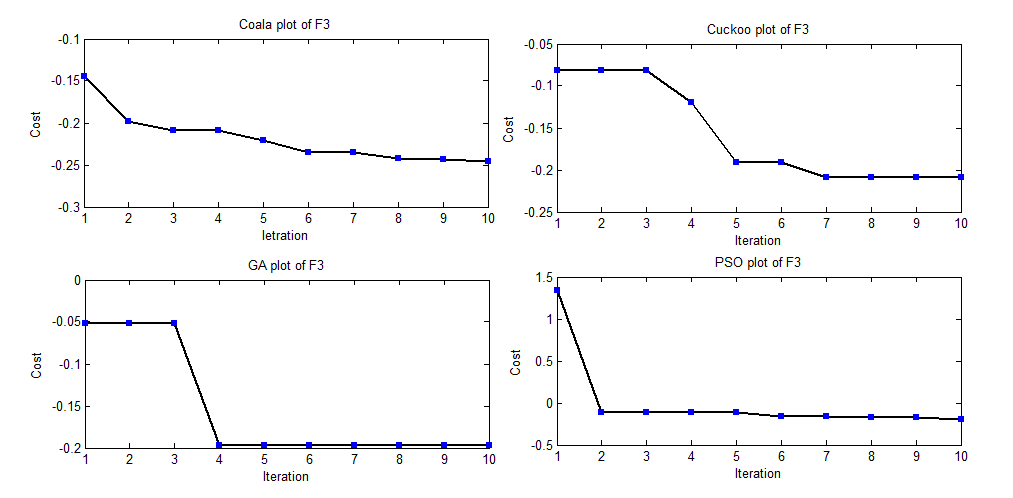
**همانطور که قابل مشاهده است الگوریتم کوآلا در اولین تکرار خود به پاسخ بهینه می رسد.**

**4-5- پیاده سازی الگوریتم بر روی تابع معیارسوم و مقایسه آن با الگوریتم های دیگر**

**محدوده متغیرهای این تابع در مقاله فاخته]1[ بی نهایت معرفی شده است اما برای اجرا و آزمایش الگوریتم ها، در این پژوهش محدوده متغیر مابین 5- و 5 در نظر گرفته شده است. تعداد جمعیت اولیه 20 و تعداد تکرار داخلی الگوریتم ها در یک بار اجرا 10 و در 30 بار اجرا برابر 100 در نظر گرفته شده است.**

**4-5-1- نتایج اعمال الگوریتم ها بر تابع معیار سوم**

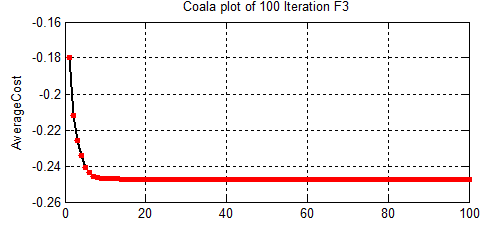
**در اجرای تصادفی الگوریتم فاخته در یک تکرار، الگوریتم فاخته در تکرار 7 به نتیجه 0.2194 - ، الگوریتم ژنتیک در تکرار 4 به پاسخ 0.19 - و الگورریتم ازدحام ذرات در تکرار 2 به 0.19923 – رسیده اند و پس از آن نتایج ثابت ماندند. این در حالی است که الگوریتم کوآلا با تنظیمات یکسان، در تکرار2 به پاسخ 0.21- و در انتها در تکرار 8 خود به پاسخ 0.2429 - می رسد.**

****

**شکل 4-10 نتایج پیاده سازی الگوریتم ها روی تابع F3**

**به منظور اثبات نتایج، الگوریتم در 30 تکرار مورد بررسی قرار گرفت که حاصل آن به شرح زیر می باشد:**

**با توجه به تنظیمات اعمال شده، در 30 بار اجرای الگوریتم فاخته به طور متوسط در تکرار39 به پاسخ 0.24- و در انتها در تکرار 100 به پاسخ 0.2467- دست یافت، الگوریتم ژنتیک در تکرار 46 به 0.2013- و در انتها در تکرار 100 به بهترین پاسخ خود یعنی 0.2141- رسید و الگوریتم ازدحام ذرات در تکرار 14 به پاسخ بهینه 0.2021- و در انتها در تکرار 52 به پاسخ 0.2271- می رسد و اما الگوریتم کوآلا به طور متوسط در تکرار 5 به 0.24- و در تکرار 12 به 0.2472- ، در الگوریتم 15 به 0.2473 و در تکرار 27 به در نقطه بهترین پاسخ موجود یعنی 0.2474- دست می یابد.**



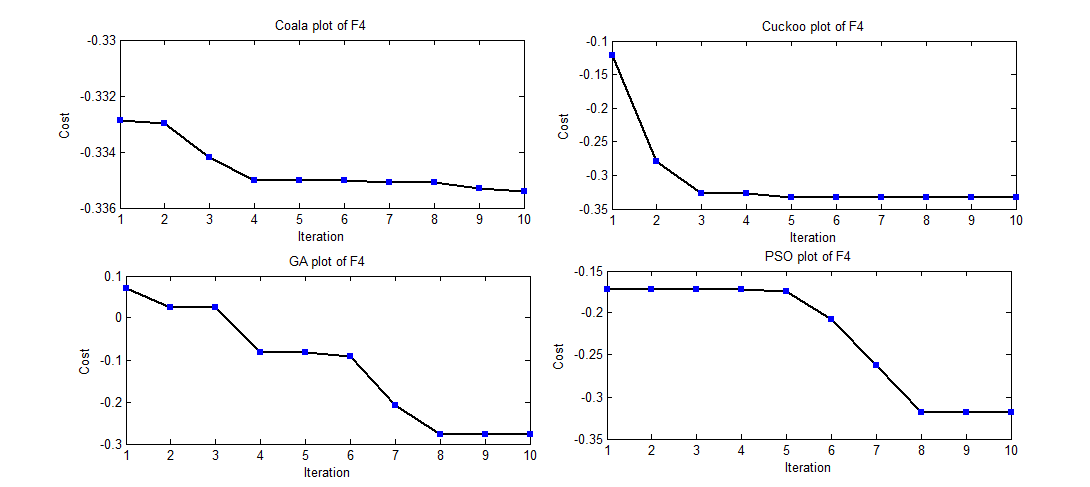
**شکل 4-11پیاده سازی الگوریتم کوآلا بر F3**

**4-6- پیاده سازی الگوریتم بر روی تابع معیار چهارم و مقایسه آن با الگوریتم های دیگر**

**همانند تابع سوم، محدوده متغیرهای تابع معیار چهارم نیز در مقاله فاخته]1[ بی نهایت بیان شده است اما برای اجرا و آزمایش الگوریتم ها، در این پژوهش محدوده متغیر مابین 5- و 5 در نظر گرفته شده است. تعداد جمعیت اولیه 20 و تعداد تکرار داخلی الگوریتم ها در یک بار اجرا 10 و در 30 بار اجرا برابر 100 در نظر گرفته شده است.**

**4-6-1- نتایج اعمال الگوریتم ها بر تابع معیار چهارم**

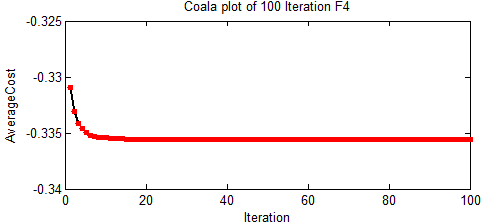
**با پیاده سازی الگوریتم فاخته، این الگوریتم در تکرار 5 به پاسخ 0.3334- دست یافت. الگوریتم ژنتیک در تکرار 8 به 0.2783 – رسیده است. همچنین الگوریتم ازدحام ذرات در تکرار 8 خود به 0.3188- رسید. با شبیه سازی الگوریتم کوآلا بر این تابع، در تکرار 4 به 0.3350- ، در تکرار 9 به 0.3353- و در تکرار 10 به 0.3354- دست یافته ایم.**

****

**شکل 4-12 نتایج پیاده سازی الگوریتم ها روی تابع F4**

**مشابه بخش های پیشین، برای اثبات تصادفی نبودن نتایج، الگوریتم ها را برای 30 بار و با 100 تکرار شبیه سازی می نماییم.**

**الگوریتم فاخته در این شبیه سازی ، به طور متوسط در تکرار 30 خود به 0.3350- و در تکرار 89 به نتیجه 0.3355- دست یافت. الگوریتم ژنتیک به طور متوسط در تکرار 17 به 0.33- ، در تکرار 89 به 0.3345- دست می یابد و از آن پس بدون تغییر می ماند. الگوریتم ازدحام ذرات به طور متوسط در تکرار 8 به 0.3188 رسیده و پس از آن تقریبا بهبودی حاصل نمی گردد. و در نهایت الگوریتم کوآلا به طور متوسط، ابتدا در تکرار 5 به 0.3350 و سپس در تکرار 11 به پاسخ بهینه یعنی 0.3355 می رسد.**



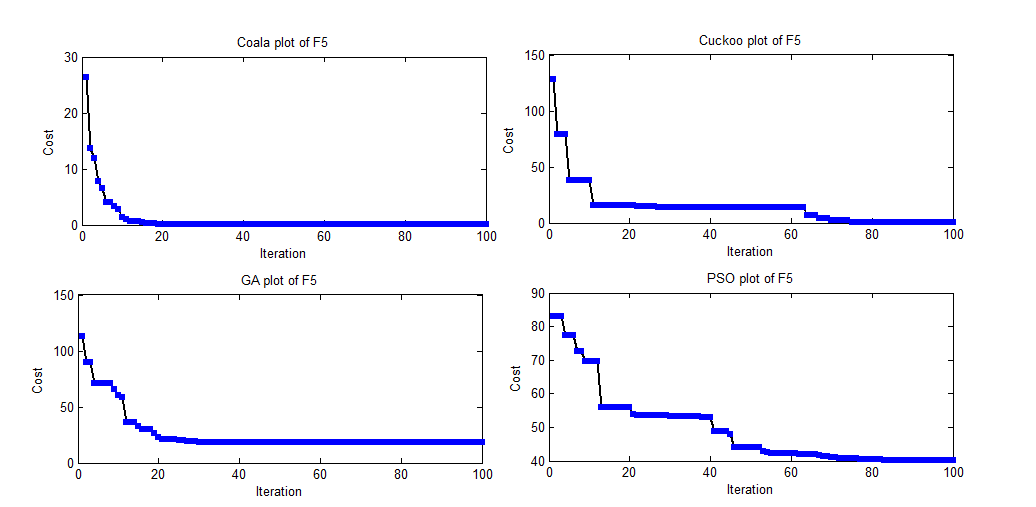
**شکل 4-13پیاده سازی الگوریتم کوآلا بر F4**

**4-7- پیاده سازی الگوریتم بر روی تابع معیار پنجم و مقایسه آن با الگوریتم های دیگر**

**در نهایت برای اثبات برتری الگوریتم کوآلا بر الگوریتم های بهینه سازی فاخته، ژنتیک و ازدحام ذرات از تابع راسترینگ 10 بعدی[[93]](#footnote-93) استفاده می نماییم. این یکی ازتوابع سخت به شمار می رود که دارای مینیم های محلی زیادی می باشد حتی درصورتی که ابعاد آن کم باشد. بنابراین این تابع می تواند معیار بسیار مناسبی برای ارزیابی عملکرد الگوریتم های بهینه سازی باشد. برای شبیه سازی الگوریتم ها بر این تابع تعداد جمعیت اولیه برابر یا 20 و تعداد تکرار حلقه داخلی در هربار اجرا برابر با 100 تنظیم می گردد.**

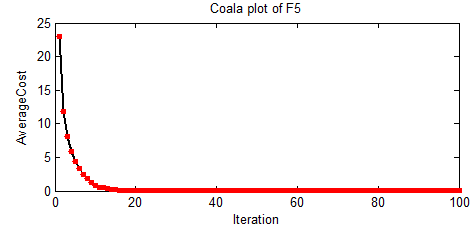
**4-7-1- نتایج اعمال الگوریتم ها بر تابع معیار پنجم**

**حاصل پیاده سازی الگوریتم فاخته بر تابع معیار پنجم، رسیدن به نقطه04 e-2.2.82 در تکرار 75 است. الگوریتم زنتیک و ازدحام ذرات همانطور که در شکل زیر پیداست به نقطه ای حتی نزدیک به صفر نیز نتوانستند دست یابند. اما الگوریتم کوآلا در تکرار 64 خود به نقطه04 e-9.9623 دست یافت.**

****

**شکل 4-14 نتایج پیاده سازی الگوریتم ها روی تابع F5**

**با توجه به پیاده سازی ها و همچنین مقاله فاخته]1[الگوریتم فاخته به طور متوسط در 66 تکرار به نقطه 0.00053169 دست می یابد. الگوریتم ژنتیک در 100 تکرار تقریبا به 15 و الگوریتم ازدحان ذرات در 100 تکرار به 29 می رسند. اما الگوریتم کوآلا به طور متوسط در تکرار 56 به نقطه ی04 e-9.8436 دست می یابد.**



**شکل 2-15 پیاده سازی الگوریتم کوآلا روی F5**

**منابع**

[7] الگوریتم‌های بهینه‌سازی فراابتکاری (همراه با کاربردهایی در مهندسی برق)/تالیف دکتر فرشاد مریخ بیات؛ انتشارات جهاد دانشگاهی؛ 1393

[8] الگوریتم‌های بهینه‌سازی فراابتکاری/تالیف مسعود یقینی، محمد رحیم اخوان کاظم‌زاده. جهاد دانشگاهی واحد صنعتی امیر کبیر [شابک](https://fa.wikipedia.org/wiki/%D8%B4%D9%85%D8%A7%D8%B1%D9%87_%D8%A7%D8%B3%D8%AA%D8%A7%D9%86%D8%AF%D8%A7%D8%B1%D8%AF_%D8%A8%DB%8C%D9%86%E2%80%8C%D8%A7%D9%84%D9%85%D9%84%D9%84%DB%8C_%DA%A9%D8%AA%D8%A7%D8%A8): [‎۹۷۸-۹۶۴-۲۱۰-۰۷۸](https://fa.wikipedia.org/wiki/%D9%88%DB%8C%DA%98%D9%87:%D9%85%D9%86%D8%A7%D8%A8%D8%B9_%DA%A9%D8%AA%D8%A7%D8%A8/978-964-210-078)

[9]بهینه‌سازی ترکیبی و الگوریتم‌های فرا ابتکاری /تالیف کورش عشقی؛ مهدی کریمی نسب انتشارات آذرین مهر.

[29]کورش عشق، مهدی کریمی نسب.، تحلیل الگوریتمها و طراحی روشهای فراابتکاری.، انتشارات دانشگاه شریف.، ۱۳۹۵.

[1] Ramin Rajabioun., Cuckoo Optimization Algorithm., Soft Computing., 5508–5518.,December 2011 .

[2] Binitha S, S Siva Sathya., A Survey of Bio inspired Optimization Algorithms., International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-2, Issue-2, May 2012

[3] David E. Goldberg., [Genetic algorithms in search, optimization, and machine learning](http://scholar.google.com/scholar?cluster=2933480113438471258&hl=en&oi=scholarr)., Machine Learning, 1989.

[4] Foulds L.R. (1981) Classical Optimization., Optimization Techniques. Undergraduate Texts in Mathematics., Springer., New York, NY.

[5] [Singiresu S. Rao](http://eu.wiley.com/WileyCDA/Section/id-302479.html?query=Singiresu+S.+Rao)., Engineering Optimization: Theory and Practice, 4th Edition.,july 2009.

[6] Talbi, El-Ghazali., Metaheuristics: From Design to Impelementation., John Wiley and 7sons 2009.

[10] Eiben, A.E., Smith, J.E., Introduction to Evolutionary Computiong., Springer 2003.

[11] Yaghini, Masoud; Akhavan., Rahim, DIMMA: A Design and Implementation Methodology for Metaheuristic Algorithms – A Perspective from Software Development., International Journal of Applied Metaheuristic Computing., Vol.1, No.4, pp. 57-74, 2010.

[12] J.H. Holland., Genetic algorithms and the optimal allocation of trials., SIAM J. Comput. 2 (2) (1973) 88–105

[13] Koza, John R., Genetic Programming: On the Programming of Computers by Means of Natural Selection., Cambridge, MA: The MIT Press.,1992.

[14] Back, T.,Evolutionary algorithms in theory and practice. Oxford University Press.,1996.

[15] John McCall., Genetic algorithms for modelling and optimization., Journal of Computational and Applied Mathematics., July 2004.

[16] Dorigo, M., Maniezzo, V., & Colorni, A., Ant System: Optimization by a colony of cooperating agents., IEEE Transactions on Systems, Man, and Cybernetics – Part B, 26, 29–41., 1996.

[17] Pham DT, Ghanbarzadeh A, Koc E, Otri S, Rahim S and Zaidi M., The Bees Algorithm., Technical Note, Manufacturing Engineering Centre., Cardiff University., UK., 2005.

[18] Kennedy, J.; Eberhart, R., [Particle Swarm Optimization.,](http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=488968) Proceedings of IEEE International Conference on Neural Networks., IV. Pp. 1942–1948.,1995.

[19] Kennedy, J. The particle swarm: social adaptation of knowledge., Proceedings of IEEE International Conference on Evolutionary Computation., pp. 303–308.,1997.

[20] Kennedy, J.; Eberhart, R.C., Swarm Intelligence., Morgan Kaufmann.,2001.

[21] Cerny, V. , A thermos dynamical approach to the traveling salesman problem: An efficient simulation algorithm., Journal of Optimization Theory and Applications., Vol. 45, pp. 41–51, 1985.

[22] Kirkpatrick, S. , Gelatt, C. D. , and Vecchi, M. P. , Optimization by simulated annealing., Science., Vol. 220, pp. 671–680 .,1983.

[23] M. Seyed Esfahani, M. Hajian Heidary , S. Jaberi., A Simulated Annealing Algorithm for Fuzzy Reliability Optimization Model in Series-Parallel and Standby Systems., International Journal of Industrial Engineering & Production Management., January 2014.

[24] [koala](http://www.thekoala.com/koala/)., TheKoala., Retrieved 4 October 2013.

[25] [Koala ecology](http://www.ehp.qld.gov.au/wildlife/koalas/koala-ecology.html)., Queensland Government: Department of Environment and Heritage Protection., Retrieved 4 October 2013.

[26] Gordon G, Menkhorst P, Robinson T, Lunney D, Martin R. & Ellis M., [Koala](http://www.iucnredlist.org/apps/redlist/details/16892)., [IUCN Red List of Threatened Species](https://fa.wikipedia.org/wiki/%D8%B3%DB%8C%D8%A7%D9%87%D9%87_%D9%82%D8%B1%D9%85%D8%B2_%DA%AF%D9%88%D9%86%D9%87%E2%80%8C%D9%87%D8%A7%DB%8C_%D8%AF%D8%B1_%D9%85%D8%B9%D8%B1%D8%B6_%D8%AE%D8%B7%D8%B1). [IUCN](https://fa.wikipedia.org/wiki/%D8%A7%D8%AA%D8%AD%D8%A7%D8%AF%DB%8C%D9%87_%D8%A8%DB%8C%D9%86%E2%80%8C%D8%A7%D9%84%D9%85%D9%84%D9%84%DB%8C_%D8%AD%D9%81%D8%A7%D8%B8%D8%AA_%D8%A7%D8%B2_%D9%85%D8%AD%DB%8C%D8%B7_%D8%B2%DB%8C%D8%B3%D8%AA) 2008., Retrieved on [2008-10-30](https://fa.wikipedia.org/w/index.php?title=2008-10-30&action=edit&redlink=1)

[27] Melanie Mitchell., An introduction to genetic algorithms., MIT Press, Cambridge, MA.,1996.

[28] Holland, John H., Adaptation in Natural and Artificial Systems., University of Michigan Press., Ann Arbor.,1975.

[30] Schmitt, Lothar M (2004), Theory of Genetic Algorithms II: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling, Theoretical Computer Science (310), pp. 181–231.

[31]Adam Lipowski , [Dorota Lipowska](http://www.sciencedirect.com/science/article/pii/S0378437111009010#!)., Roulette-wheel selection via stochastic acceptance., [Physica A: Statistical Mechanics and its Applications](http://www.sciencedirect.com/science/journal/03784371).,12,2011.

[32] Gillbert Syswerda., Uniform Crossover in Genetic Algorithm.,3rd International Conference on Genetic Algorithm.,1989.

[33] Tomasz D. Gwiazda, Genetic Algorithms Reference Vol.1 Crossover for single-objective numerical optimization problems, Tomasz Gwiazda, Lomianki, 2006. [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [83-923958-3-2](https://en.wikipedia.org/wiki/Special:BookSources/83-923958-3-2).

[34] Claudio Comis Da Ronco, Ernesto Benini, GeDEA-II: a simplex-crossover based multi objective evolutionary algorithm including the genetic diversity as objective, GECCO’12 Companion, July 7–11, 2012, Philadelphia, PA, USA. ACM 978-1-4503-1178-6/12/07.

[35] [Ali R.Yildiz](http://www.sciencedirect.com/science/article/pii/S0020025512002046" \l "!)., A comparative study of population-based optimization algorithms for turning operations.,Information Science.,November 2012.

[36] Crepinsek, M., Liu, S.-H., Mernik, M., Exploration and exploitation in evolutionary algorithms: A

survey., ACM Comput. Surv. 45, 3, Article 35.,June 2013.

[37] [AH Gandomi](https://scholar.google.com/citations?user=VMf3wfMAAAAJ&hl=en&oi=sra), [XS Yang](https://scholar.google.com/citations?user=fA6aTlAAAAAJ&hl=en&oi=sra), [AH Alavi](https://scholar.google.com/citations?user=Zp20Gw0AAAAJ&hl=en&oi=sra)., [Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems](http://link.springer.com/article/10.1007/s00366-011-0241-y).,Engineering with computers.,Springer-2013.

[38] M Yagiura, T Ibaraki .,[On metaheuristic algorithms for combinatorial optimization problems](http://onlinelibrary.wiley.com/doi/10.1002/1520-684X(200103)32:3%3C33::AID-SCJ4%3E3.0.CO;2-P/full)., Systems and Computers in Japan., 2001.

[39] [WJ Gutjahr](https://scholar.google.com/citations?user=DJ7D-YwAAAAJ&hl=en&oi=sra)., [ACO algorithms with guaranteed convergence to the optimal solution](http://www.sciencedirect.com/science/article/pii/S0020019001002587)., Information processing letters., Elsevier.,2002.

[40] [M Gendreau](https://scholar.google.com/citations?user=CilK624AAAAJ&hl=en&oi=sra), JY Potvin., [Metaheuristics in combinatorial optimization](http://link.springer.com/article/10.1007/s10479-005-3971-7).,Annals of Operations Research.,Springer.,2005.

[41] A. Ahrari, M. Shariat-Panahi, A.A. Atai, G.E.M., a novel evolutionary optimization method with improved neighborhood search., Applied Mathematics and Computation., 2009.

[42] Lu Hong., A novel particle swarm optimization method using clonal selection algorithm., International Conference on Measuring Technology and Mechatronics Automation., vol. 2, 2009, pp.471–474.

[43] V. Chellaboina, M.K. Ranga., Reduced order optimal control using genetic algorithms., American Control Conference., Portland, USA, vol. 2, 2005.

[44] D.B. Fogel., An evolutionary approach to the traveling salesman problem., Biological Cybernetics 60., 2004.

[45] B. Bontoux, D. Feillet., Ant colony optimization for the traveling purchaser problem., Computers & Operations Research 35.,2008.

[46] R.L. Johnston, H.M. Cartwright., Applications of Evolutionary Computation in Chemistry., Springer-Verlag., Berlin., Heidelberg., 2004.

[47] C.-C. Wu, K.-C. Lai, R.-Y. Sun., GA-based job scheduling strategies for fault tolerant grid systems., IEEE Asia-Pacific Services Computing Conference., 2008.

[48] L.V. Arruda, M.C. Swiech, M.R. Delgado, F. Neves Jr., PID control of MIMO process based on rank niching genetic algorithm., Applied Intelligence 29., 2008.,

[49] H.A. Varol, Z. Bingul., A new PID tuning technique using ant algorithm., Proceeding of the IEEE 2004 American Control Conference., vol. 3, 2004.

[50] H.H. Balaci, J.F. Valenzuela., Scheduling electric power generators using particle swarm optimization combined with Lagrangian relaxation method., International Journal of Applied Mathematics and Computer Science 14.,2004.

[51] C. Andrieu, A. Doucet., Simulated annealing for maximum a posteriori parameter estimation of hidden Markov models., IEEE Transactions on Information Theory 46., 2000.

[52] E. Atashpaz-Gargari, C. Lucas, Imperialist Competitive Algorithm: An algorithm for optimization inspired by imperialistic competition, IEEE Congress on Evolutionary Computation, 2007.

[53] اسماعیل آتش پز گرگری، «توسعه الگوریتم بهینه‌سازی اجتماعی و بررسی کارایی آن»، پایان‌نامه کارشناسی ارشد دانشکده مهندسی برق و کامپیوتر دانشگاه تهران، ۱۳۸۷.

[54] ندا وحید زاده، مهدی جوانمرد.، بازنگری بر الگوریتم های الهام گرفته شده طبیعی: الگوریتم شب تاب، الگوریتم خفاش و الگوریتم جستجوی فاخته، اولین همایش ملی پژوهش های مهندسی رایانه، 1393.

[55] Emad Elbeltagi, Tarek Hegazy, Donald Grierson, Comparison among five Evolutionary-based Optimization Algorithms, Advanced Engineering Informatics,2005.

1. Optimization [↑](#footnote-ref-1)
2. Meta-Heuristic Algorithm [↑](#footnote-ref-2)
3. Koala [↑](#footnote-ref-3)
4. Minimization [↑](#footnote-ref-4)
5. Maximization [↑](#footnote-ref-5)
6. Fitness Function [↑](#footnote-ref-6)
7. Cost Function [↑](#footnote-ref-7)
8. Objective Function [↑](#footnote-ref-8)
9. Classical Optimization [↑](#footnote-ref-9)
10. Artificial Intelligent [↑](#footnote-ref-10)
11. Continuous Function [↑](#footnote-ref-11)
12. Differential Calculus [↑](#footnote-ref-12)
13. Single-Variable Function [↑](#footnote-ref-13)
14. Multivariable Function with no Constraints [↑](#footnote-ref-14)
15. Multivariable Function with Constraints [↑](#footnote-ref-15)
16. Heuristic Optimization Algorithms [↑](#footnote-ref-16)
17. Meta-Heuristic [↑](#footnote-ref-17)
18. Hyper Meta-Heuristic [↑](#footnote-ref-18)
19. Convergence [↑](#footnote-ref-19)
20. Performance [↑](#footnote-ref-20)
21. Genetic Algorithm [↑](#footnote-ref-21)
22. Cuckoo Optimization Algorithm [↑](#footnote-ref-22)
23. Ant Colony Algorithm [↑](#footnote-ref-23)
24. Bees Algorithm [↑](#footnote-ref-24)
25. Particle Swarm Optimization [↑](#footnote-ref-25)
26. Simulated Annealing [↑](#footnote-ref-26)
27. John Holland [↑](#footnote-ref-27)
28. Global Search [↑](#footnote-ref-28)
29. Charles Darwin Theory [↑](#footnote-ref-29)
30. Crossover [↑](#footnote-ref-30)
31. Mutation [↑](#footnote-ref-31)
32. Fitness [↑](#footnote-ref-32)
33. Traditional Search [↑](#footnote-ref-33)
34. Single-Point Crossover [↑](#footnote-ref-34)
35. Multi-Point Crossover [↑](#footnote-ref-35)
36. Uniform Crossover [↑](#footnote-ref-36)
37. Replacement [↑](#footnote-ref-37)
38. Reversal [↑](#footnote-ref-38)
39. Dorigo [↑](#footnote-ref-39)
40. Approximate Solution [↑](#footnote-ref-40)
41. Pheromone [↑](#footnote-ref-41)
42. Positive Feedback [↑](#footnote-ref-42)
43. Distributed Computation [↑](#footnote-ref-43)
44. Constructive Greedy Heuristic [↑](#footnote-ref-44)
45. Premature Convergence [↑](#footnote-ref-45)
46. Dynamic [↑](#footnote-ref-46)
47. Pham [↑](#footnote-ref-47)
48. Scout Bees [↑](#footnote-ref-48)
49. Waggle Dance [↑](#footnote-ref-49)
50. Locl Search [↑](#footnote-ref-50)
51. Global Search [↑](#footnote-ref-51)
52. Clustering System [↑](#footnote-ref-52)
53. Manufacturing [↑](#footnote-ref-53)
54. Control [↑](#footnote-ref-54)
55. Multi-Objective Optimization [↑](#footnote-ref-55)
56. Particle Swarm [↑](#footnote-ref-56)
57. Kennedy [↑](#footnote-ref-57)
58. Non-Linear Function [↑](#footnote-ref-58)
59. Krik Patrik [↑](#footnote-ref-59)
60. Annealing [↑](#footnote-ref-60)
61. Threshold [↑](#footnote-ref-61)
62. Accept Probability [↑](#footnote-ref-62)
63. Cycle specific [↑](#footnote-ref-63)
64. Imperialist Competitive Algorithm [↑](#footnote-ref-64)
65. Country [↑](#footnote-ref-65)
66. Colony [↑](#footnote-ref-66)
67. Empire [↑](#footnote-ref-67)
68. Imperialistic Competition [↑](#footnote-ref-68)
69. Revolution [↑](#footnote-ref-69)
70. Firefly Algorithm [↑](#footnote-ref-70)
71. Xin-She Yang [↑](#footnote-ref-71)
72. Luciferin [↑](#footnote-ref-72)
73. Ramin Rajabioun [↑](#footnote-ref-73)
74. Eucalyptus [↑](#footnote-ref-74)
75. Initial Population [↑](#footnote-ref-75)
76. Feeding [↑](#footnote-ref-76)
77. Chromosome [↑](#footnote-ref-77)
78. Particle Position [↑](#footnote-ref-78)
79. Habitat [↑](#footnote-ref-79)
80. Profit Function [↑](#footnote-ref-80)
81. Selection [↑](#footnote-ref-81)
82. Roulette Wheels [↑](#footnote-ref-82)
83. Elitist [↑](#footnote-ref-83)
84. Scaling [↑](#footnote-ref-84)
85. Tournament [↑](#footnote-ref-85)
86. Fitness Proportionate Selection [↑](#footnote-ref-86)
87. Crossover [↑](#footnote-ref-87)
88. Reproduction [↑](#footnote-ref-88)
89. Parent Feeding [↑](#footnote-ref-89)
90. Children Feeding [↑](#footnote-ref-90)
91. Exploration [↑](#footnote-ref-91)
92. Exploitation [↑](#footnote-ref-92)
93. 10-Dimensial Rastrigin Function [↑](#footnote-ref-93)