

SID



سرویس های ویژه



سرویس ترجمه تخصصی



کارگاه های آموزشی



بلاگ مرکز اطلاعات علمی



عضویت در خبرنامه



فیلم های آموزشی

کارگاه های آموزشی مرکز اطلاعات علمی جهاد دانشگاهی



PROPOSAL

پروپوزال

مركز آموزش
پروپوزال نویسی و پایان نامه نویسی

کارگاه آنلاین
پروپوزال نویسی و پایان نامه نویسی



مركز آموزش
روش تحقیق و مقاله نویسی علوم انسانی

کارگاه آنلاین
روش تحقیق و مقاله نویسی علوم انسانی



ISI
Scopus

مركز آموزش
آشنایی با پایگاه های اطلاعات علمی بین المللی و ترکیه های جستجو

کارگاه آنلاین آشنایی با پایگاه های اطلاعات علمی بین المللی و ترکیه های جستجو

الگوریتم ژنتیک گروهی مبتنی بر الگوریتم پرندگان

محمد علی چاووشیان

آموزشکده فنی و حرفه ای ملاحدرا - رامسر

Mohammad Ali Chavooshian

alichavooshian@yahoo.com

چکیده

هدف این مقاله ارائه و ارزیابی یک الگوریتم بهینه سازی جدید است. الگوریتم جدید، الگوریتم ژنتیک پرندگان نام دارد. این الگوریتم نوعی ترکیبی از الگوریتم ژنتیک و الگوریتم بهینه سازی ذرات است. این مقاله به نقاط قوت و ضعف دو الگوریتم می پردازد. سپس چگونگی ترکیب شدن ویژگی های هر دو را شرح داده و جزئیات الگوریتم را بیان می کند. هر سه الگوریتم با استفاده از هشت مسئله بهینه سازی استاندارد ادبیات موضوع با هم مقایسه می شوند. نشان داده می شود که الگوریتم ژنتیک پرندگان، کارایی برتری در ۷۵٪ حالات تست شده دارد. در ۲۵٪ بقیه حالات، کارایی بیشتری نسبت به الگوریتم ژنتیک یا بهینه سازی ذرات داشته و در هیچ حالتی بدتر از دو الگوریتم دیگر نیست. بهبودهای ممکن در آینده نیز به طور خلاصه بررسی می شوند.

واژگان کلیدی: الگوریتم ژنتیک، الگوریتم پرندگان، شباهت ها، بهینه سازی ذرات

۱- مقدمه

این مقاله الگوریتم ژنتیک گروهی را معرفی می کند. این الگوریتم ترکیبی از الگوریتم های ژنتیک و بهینه سازی ذرات است. هر سه الگوریتم از طریق مسائل استاندارد بهینه سازی ارزیابی می شوند.

الگوریتم ژنتیک که در اواسط دهه ۷۰ توسط **John Holland** به شهرت رسید [۱]، تئوری های زیستی تکامل، ژنتیک، انتخاب طبیعی و جهش را در یک الگوریتم رایانشی ادغام می کند که به طور اکتشافی راه حلی بهینه را برای یک مسئله جستجو می کند. راه حلی ممکن برای یک مسئله توسط یک ژنوم نشان داده می شود که اصولاً شامل یک کروموزوم است. این کروموزوم شامل مقادیری با نام ژن است. جهش و تقاطع نیز برای تغییر کروموزوم ها استفاده می شود. جهش شامل تغییرات کوچک تصادفی در کروموزوم هاست. تقاطع شامل تبادل میزانی از ویژگی های ژنتیکی می باشد.

در یک الگوریتم ژنتیک (**GA**)، جمعیتی از کروموزوم ها به صورت زیر تکامل می یابند. جمعیت به طور تصادفی ایجاد می شود. نسل های جدید از طریق انتخاب مکرر برانزده ترین افراد و جهش و تقاطع آنها تولید می شوند. این فرآیند تا زمانی که یکی از کروموزوم ها به برانزده گی مطلوب رسیده یا تا تعداد محدودی از نسل ها ادامه می یابد. جزئیات و تغییرات این الگوریتم در تحقیقات زیادی مورد بررسی قرار گرفته است [۲].

اخیراً، **James Kennedy** و **Russell Eberhart** الگوریتم بهینه سازی اکتشافی جالبی طراحی کرده اند که بهینه سازی دسته ذرات نام دارد [۳]. جزئیات و تغییرات جدید این الگوریتم نیز در [۲] آورده شده است. مشابه با الگوریتم ژنتیک، این الگوریتم نیز از پدیده های زیستی الهام گرفته شده است.

در بهینه سازی ذرات (**PSO**)، دسته ای از ذرات مورد استفاده قرار گرفته است. هر ذره، راه حلی ممکن را در فضای جستجو نشان می دهد و مکان هر ذره بر طبق تجربه خود و همسایگانش تنظیم می شود تا بهترین مقدار یک تابع مشخص به دست آید.

در دومین بخش این مقاله، الگوریتم ژنتیک پرندگان را معرفی می کنیم. در سومین بخش، هر سه الگوریتم را با استفاده از مسائل بهینه سازی استاندارد ادبیات موضوع مقایسه کرده و در بخش نهایی به نتیجه گیری و راهکاری آینده الگوریتم ژنتیک پرندگان (**GFA**) پرداخته می شود.

۲- الگوریتم ژنتیک پرندگان

۲-۱ - مقایسه PSO و GA

۲-۱-۱ شباهت‌ها

ارتباط واضحی بین PSO و GA وجود دارد که ترکیب این دو الگوریتم را ممکن می‌سازد. آنها هر دو شامل دسته‌هایی از راه حل‌های مسئله مورد نظر هستند. معمولاً کروموزوم‌ها رشته‌های بیتی هستند. ولی ممکن است آرایه‌هایی از اعداد حقیقی نیز باشند. از طرف دیگر، یک ذره در PSO در یک فضای چندبعدی مکانی دارد که معمولاً مجموعه‌ای از اعداد حقیقی است. با نوعی تغییر، می‌توان مکان را با مجموعه‌ای از بیت‌ها مشخص کرد [۲]. در این مقاله، هم کروموزوم‌ها و مکان‌های ذرات را آرایه‌هایی از اعداد حقیقی در نظر می‌گیریم. کروموزوم‌ها متناظر با مکان‌های ذرات هستند. از آنجا که کروموزوم‌ها در هر نسل تغییر می‌کنند، این نسل‌ها متناظر با گام‌های زمانی گسسته هستند که در آنها، مکان‌های ذرات تغییر می‌کنند.

تابع برازندگی الگوریتم ژنتیک متناظر با تابعی است که در الگوریتم PSO باید بهینه شود.

۲-۱-۲ تفاوت‌ها

تفاوت اصلی بین دو الگوریتم به نحوه تغییر افراد مرتبط است. کروموزوم‌ها از طریق جهش و تقاطع تصادفی تغییر می‌کنند. ولی هر ذره از طریق نوسان تصادفی بین دو مرکز جذابیت تغییر می‌کند که یکی از این مراکز در بهترین مکان ذره تاکنون است. این امر معمولاً مولفه شناختی نامیده می‌شود؛ چرا که برپایه دانش تجربی ذره است [۲]. مرکز دیگر یک ذره بهترین مکان یافته شده توسط ذرات در همسایگی آن است. این امر معمولاً مولفه اجتماعی نامیده می‌شود؛ چرا که برپایه دانش تبادل شده در اجتماع است. اگر همسایگی یک ذره کل دسته باشد، این روش بهترین سراسری نامیده می‌شود. اگر همسایگی ذره محدودتر باشد، بهترین محلی نام دارد.

بدیهی است که GA تصادفی‌تر از PSO می‌باشد. اگر والدین را با فرزندان شناسایی کنیم، ممکن است کروموزوم لزوماً از طریق جهش یا تقاطع بهبود نیابد. ممکن است برازندگی بدتری داشته باشد. حتی ممکن است برازندگی بهترین کروموزوم از یک نسل به نسل دیگر تقلیل یابد. بدین دلیل، الگوریتم‌های ژوتیک معمولاً از نخبه‌گرایی استفاده می‌کنند. در این روش باید چند فردی که در نسلی بهترین هستند، تا نسل بعدی بدون تغییر باقی بمانند PSO. آنها نیازی به نخبه‌گرایی ندارند؛ چرا که بهترین‌های قبلی هیچگاه از دست داده نمی‌شوند.

به دلایلی، PSO مشابه با نسخه‌های پیشرفته‌تر تپه‌نوردی هستند. مقادیری که قبل پیدا شده‌اند، به شدت جستجو را تحت تاثیر قرار می‌دهند. در GA، مقادیر قبلی تنها بنیانی هستند که تغییرات از آنها نشأت می‌گیرند. آنها ذات تغییرات را تحت تاثیر قرار نمی‌دهند. اگرچه ممکن است تقاطع در GA منجر به ترکیب راه‌های جزئی شود، ولی گاهی آن‌ها را به هم می‌ریزد.

انتظار ما این است که PSO از الگوریتم ژنتیک سریع‌تر باشد، ولی به دلیل اینکه ممکن است توسط تابعی به اندازه کافی نامنظم به تله بیفتند، مقاومت کمتری دارد.

۳-۱-۲ - ژنتیک پرندگان

باور داریم به کارگیری GA و PSO در هرگام یک الگوریتم غیرضروری بوده و حتی زیان آور است. درمقابل ترکیبی از الگوریتم ژنتیک و PSO طراحی کرده ایم که فازهای GA و PSO جداگانه ای دارد. ما این الگوریتم را ژنتیک پرندگان نامیده ایم. هدف اصلی این تحقیق این است که مزایای چنین روش ترکیبی با فازهای GA و PSO جداگانه نمایش داده شود. بدین منظور آن را با الگوریتم های دیگر مقایسه می کنیم.

ژنتیک پرندگان از طریق اجرای فاز الگوریتم ژنتیک در تعداد محدودی تکرار و سپس اجرای فاز بهینه سازی ذرات در تعداد محدودی تکرار عمل می کند. تعداد تکرار در هر فاز قابل تنظیم است، ولی به طور پیش فرض هر یک ۵۰ تکرار در نظر گرفته می شود. این فرآیند تا زمانی که حداکثر تعداد تکرار انجام گیرد، تکرار می شود.

فاز GA از انتخاب مسابقه ای با تقاطع تک نقطه ای استفاده می کند. جهش ها افزایشنده بوده و از یک توزیع گاوسی با انحراف معیار کسری ثابت (معمولاً ۰.۰۱) از بهینه سازی انتخاب می شود. نرخ جهش پیشفرض ۰.۱۰ است.

در فاز PSO، ذرات در مجموعه هایی با نام همسایگی با اندازه پیشفرض ۷ گروه بندی می شوند. ذرات براساس اینرسی، نیرویی فنر مانند و تصادفی به سمت بهترین مکان قبلا ملاقات شده و نیرویی مشابه به سمت بهترین مکان ملاقات شده توسط اعضای در همسایگی اش حرکت می کند. هر بار که ذرات از مکان به مکان دیگری حرکت می کنند، نیروی فنرمانند پیرامون همسایگی و ثابت شده و بهترین مکان های شخصی با درصدی تصادفی اصلاح می شوند. حداکثر سرعت ذرات از نیم فاصله بین محدوده های ذرات تجاوز نمی کند. به بیان دیگر یک ذره نباید گامی بزرگتر از نیمی از کل فضای جستجویی کند.

۴-۱-۲ - تحقیقات دیگر

Eberhart and Shi [4] شباهت های بین الگوریتم ژنتیک و بهینه سازی ذرات را بررسی کرده و تاکید می کنند عناصر یک الگوریتم باید در الگوریتم دیگر به کار گرفته شوند. همزمان با این پیشنهادیه، Castelli, Manzoni, and Vanneschi [5] نویسندگانی را مرجع قرار می دهند که از عناصر حافظه بهینه سازی ذرات در الگوریتم ژنتیک استفاده کرده اند. همچنین Engelbrecht [2] به نویسندگانی اشاره می کند که جهش را در الگوریتم بهینه سازی ذرات وارد کرده اند.

مشابه ترین تحقیق با تحقیق ما [6] Juang است. در تحقیق او، هم الگوریتم ژنتیک و هم بهینه سازی ذرات در یک الگوریتم ادغام شده اند. Juang از این الگوریتم برای انتساب وزن یک شبکه عصبی (نوعی الگوریتم اکتشافی دیگر) استفاده کرده و الگوریتم خود را برپایه کارآیی شبکه عصبی می سنجد. در الگوریتم دوگانه او، Juang بخش بهینه سازی ذرات را تنها روی نیمه بهتر کروموزوم های الگوریتم ژنتیک (با نام نخبه های بهبودیافته) اجرا می کند. سپس از این نخبه ها در انتخاب، تقاطع و جهش استفاده می نماید. فرزندان ایجاد شده، نیمی از نسل بعدی را تشکیل داده و نخبه های بهبودیافته نیمه دیگر جمعیت خواهند بود.

از دیدگاه ما، حتی الگوریتم Juang نیز بیش از حد پیچیده است. هدف ما در این تحقیق این است که نشان دهیم یک الگوریتم چندفازی ساده برای جبران کاستی های GA و PSO کافی می باشد.

۳- آزمایشات

در کل هشت مسئله برای ارزیابی کارایی سه الگوریتم مورد استفاده قرار گرفت. این مسائل از مسائلی ساده گرفته تا پیچیده انتخاب شده اند تا به ت صوری شفاف از چگونگی عملکرد هر الگوریتم دست یافته شود. این مسائل، مسائل ارزیابی الگوریتم معیار هستند که با هدف تست کارایی الگوریتم های بهینه سازی طراحی شده اند [۷، ۸، ۹]. مسائل زیر مورد استفاده قرار گرفته اند:

1. Spherical $f(\bar{x}) = \sum_{j=1}^n x_j^2$ where $-100 \leq x_j \leq 100$
2. Ackley $f(x) = -20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{j=1}^n x_j^2}) - \exp(\frac{1}{n} \sum_{j=1}^n \cos 2\pi x_j) + 20 + e$ where $-30 \leq x_j \leq 30$
3. Griewangk $f(\bar{X}) = -\sum_{j=1}^n \frac{x_j^2}{4000} - \prod_{j=1}^n \cos(\frac{x_j}{\sqrt{j}}) + 1$ where $-500 \leq X_j \leq 500$
4. Michalewicz $f(\bar{X}) = -\sum_{j=1}^n \sin(X_j) \left(\sin(\frac{jX_j^2}{\pi})\right)^{2m}$ where $m=10, 0 \leq X_j \leq \pi$
5. Rastrigin $f(\bar{X}) = 10n + \sum_{j=1}^n (x_j^2 - 10 \cos(2\pi x_j))$ where $-5.12 \leq x_j \leq 5.12$
6. Rosenbrock $f(\bar{X}) = \sum_{j=1}^{n/2} (100(X_{2j} - X_{2j-1}^2)^2 + (x_{2j-1} - 1)^2)$ where $-2.048 \leq x \leq 2.0487$
7. Schwefel $f(\bar{X}) = 418.9829n - \sum_{j=1}^n x_j \sin(\sqrt{|x_j|})$ where $-500 \leq x_j \leq 500$
8. Shekel $f(\bar{X}) = \sum_{j=1}^{10} [\sum_{i=1}^4 (x_i - c_{ij})^2 + \beta_j]$ where $-10 \leq x_j \leq 30$

Where $\bar{\beta} = \frac{1}{10} [1 \ 2 \ 2 \ 4 \ 4 \ 6 \ 3 \ 7 \ 5 \ 5]$

$$C = \begin{bmatrix} 4 & 1 & 8 & 6 & 3 & 2 & 5 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 5 & 1 & 2 & 3.6 \\ 4 & 1 & 8 & 6 & 3 & 2 & 3 & 8 & 6 & 7.0 \\ 4 & 1 & 8 & 6 & 7 & 9 & 3 & 1 & 2 & 3.6 \end{bmatrix}$$

هدف حداقل ساختن f است. حداقل مقدار برای هر f صفر به دست آمد. در هر حالت، مقدار $n=10$ در روابط فوق مورد استفاده قرار گرفت. اندازه جمعیت براساس آزمایش و خطا ۵۶ انتخاب شد. به دلیل اینکه $n=10$ ، هر الگوریتم به ازای هر مسئله ۵۰ بار اجرا شد. هر اجرای الگوریتم برای هر مسئله شامل ۵۰۰۰۰ گام بود. بعد از اجرا، بهترین کارایی الگوریتم ذخیره شد. سپس به ازای هر الگوریتم، میانگین این مقادیر در تمام اجراها در نظر گرفته شد. این روش تحلیل متوسط کارایی هر الگوریتم را براساس هر مسئله ممکن می سازد.

۳-۱- رفتار پویای معمول

اجراهای که رفتار زمانی معمولی نشان دادند در ادامه آورده شده‌اند. به دلیل محدودیت در فضا، تنها خلاصه‌ای از نتایج در هر حالت ارائه شده است. برای جزئیات بیشتر به Brooks [۱۰] مراجعه نمایید.

برای مسئله کروی، الگوریتم بهینه‌سازی ذرات شروع خوبی داشته و در جهتی حرکت می‌کند که دو الگوریتم دیگر موفق به دستیابی به آن نشدند. اکثر بهبودهای پیشنهادی در الگوریتم ژنتیک پرندگان، هنگامی که در حالت بهینه‌سازی ذرات عمل می‌کرد، تحقق یافت که باعث شد از الگوریتم ژنتیک فاصله بگیرد. هردو الگوریتم ژنتیک و پرندگان در طول تکرارهای متمادی بهبود یافتند. ولی هیچ یک قادر به رسیدن به الگوریتم بهینه‌سازی ذرات نبودند.

در مسئله ackley، الگوریتم بهینه‌سازی ذرات شروعی سریع داشت، ولی به زودی در بهینه محلی گیر کرده و بهبود دیگری نداشت. این امر در صورتی اتفاق می‌افتد که کل جمعیت به بهینه‌ای محلی محدود شوند و بهترین مکان‌ها (هم در همسایگی هم شخصی) همه در محدوده آن بهینه قرار بگیرند. دو الگوریتم دیگر در طول تکرارهای متمادی بهبود یافتند. الگوریتم پرندگان پیشرفت بیشتری نسبت به ژنتیک داشت. هردو الگوریتم به نتیجه کمتری (بهینه تری) نسبت به بهینه‌سازی ذرات رسیدند و پرندگان کمترین نتیجه را به دست آورد.

در رابطه Griewangk، الگوریتم بهینه‌سازی ذرات پیشرفت سریعی دارد. پرندگان هنگامی که وارد حالت بهینه‌سازی ذرات می‌شود، فاصله خود را میان دو الگوریتم دیگر حفظ می‌کند. سپس قادر است بیشتر از الگوریتم بهینه‌سازی ذرات پیشروی کند. الگوریتم ژنتیک بهبودی پیوسته نشان می‌دهد؛ ولی قادر به رسیدن به دو الگوریتم دیگر نیست.

الگوریتم بهینه‌سازی ذرات در مسئله Michalewicz، در مقایسه با پرندگان و ژنتیک ضعیف عمل می‌کند. درحالی‌که الگوریتم بهینه‌سازی ذرات در طول زمان بهبود می‌یابد، دو الگوریتم دیگر پیشرفت بسیار سریعتری دارند. پرندگان و ژنتیک نزدیک به هم پیشروی می‌کنند، ولی پرندگان جلوتر رفته و مقدار کمتری نسبت به ژنتیک پیدا می‌کند.

نتایج رابطه Rastrigin نیز مشابه با Michalewicz است. بهینه‌سازی ذرات در طول تکرارها بهبود می‌یابد؛ ولی بهبود آن در مقایسه با دو الگوریتم دیگر بسیار کند است. بین دو الگوریتم دیگر، پرندگان قادر است به مقدار میانگین کمتری دست یابد.

در رابطه RosenBrock، بهینه‌سازی ذرات پیشرفت سریعی دارد. ژنتیک و پرندگان در طول تکرارهای متمادی مشابه هم عمل می‌کنند. در نهایت پرندگان می‌تواند به بهینه‌سازی ذرات رسیده و از آن جلو بزنند. ژنتیک خط سیر بهبود یکنواختی دارد؛ ولی قادر به رسیدن به دو الگوریتم دیگر نیست.

در مسئله Schwefel، کارآیی بهینه‌سازی ذرات به شدت کمتری نسبت به دو الگوریتم دیگر نشان داد. برای درک دلیل این امر، اجراهای بیشتری با بازه‌های کوچکتری از مقادیر ممکن گرفته شد که تحت این اجراها بهینه‌سازی ذرات توانست به نسبت بهتر عمل کند. کارآیی هر سه الگوریتم در رابطه Shekel نزدیک به هم بود. بهینه‌سازی ذرات کندتر از همه شروع کرده و پرندگان هنگامی که از ژنتیک به بهینه‌سازی ذرات تغییر حالت می‌دهد، از همه جلو می‌افتد. پس از این، پرندگان به بهترین کارآیی دست می‌یابد. ژنتیک و بهینه‌سازی ذرات در اوایل با هم رقابت می‌کنند، ولی همانطور که تکرارهای بیشتری انجام می‌شود، ژنتیک می‌تواند به کارآیی بهتری نسبت به بهینه‌سازی ذرات دست یابد.

۳-۲ - مقایسه کلی

نتایج کلی در جدول زیر آمده است که حاوی کمترین مقدار متوسط برگردانده شده در همه اجراهای الگوریتم ها می باشد. مقایسه مقادیر متوسط نشان می دهد کدام الگوریتم به طور میانگین بهتر عمل می کند. به منظور تفسیر ساده تر جدول، بهترین مقادیر پررنگ و بدترین مقادیر با فونت ایتالیک نشان داده شده اند.

Function	Genetic Algorithm	Genetic Flock	PSO
Spherical	<i>1.74 x 10⁻³</i>	<i>2.00 x 10⁻¹⁹</i>	0
Ackley	<i>1.56 x 10⁻³</i>	<i>1.40 x 10⁻¹⁰</i>	<i>6.67 x 10⁻¹</i>
Griewangk	<i>5.45 x 10⁻²</i>	<i>5.90 x 10⁻³</i>	<i>4.30 x 10⁻²</i>
Michalewicz	<i>4.89 x 10⁻⁷</i>	0	<i>4.06 x 10⁻¹</i>
Rastrigin	<i>8.66 x 10⁻⁶</i>	<i>1.71 x 10⁻¹⁵</i>	<i>4.29</i>
RosenBrock	<i>7.73 x 10⁻³</i>	<i>9.04 x 10⁻¹⁵</i>	<i>3.66 x 10⁻³</i>
Schwefel	<i>5.38 x 10⁻³</i>	<i>1.27 x 10⁻⁴</i>	<i>1.16 x 10³</i>
Shekel	<i>2.21 x 10⁻¹</i>	<i>2.21 x 10⁻¹</i>	<i>2.23 x 10⁻¹</i>

جدول ۱. کمترین مقدار متوسط برای همه مسائل آزمایشی

در شش مسئله از هشت مسئله، الگوریتم پرندگان بهترین میانگین کارآیی را نشان داد. در مسئله ای دیگر، الگوریتم دیگری نیز بهترین کارآیی یکسانی داشته و در سایر مسائل دومین الگوریتم برتر بود. الگوریتم ژنتیک در یک مسئله بهترین کارآیی را داشت، در چهار رابطه دومین الگوریتم برتر بود و در سه رابطه دیگر سومین رتبه کارآیی را داشت. الگوریتم بهینه سازی ذرات بهترین میانگین کارآیی را در یک مسئله داشت، در دو مسئله دومین الگوریتم برتر بود و در سایر مسائل سومین رتبه را داشت. در کل این نتایج انتظارات ما را در مورد کارآیی این سه الگوریتم برآورده کردند.

۴- نتیجه گیری

در کل، در نتایج مشاهده می شود که به ازای مسائل در نظر گرفته شده، الگوریتم ژنتیک پرندگان بهترین کارایی را دارد. در نهایت، این الگوریتم دارای بالاترین کارایی ۷۵٪ اندازه گیری ها بود. در ۲۵٪ باقیمانده، پرندگان از یک الگوریتم دیگر کارایی بالاترین نشان داد. پرندگان تنها الگوریتمی بود که در هیچ اجرایی بدترین کارایی را نداشت. مطابق با ویژگی های معمول بهینه سازی ذرات، این الگوریتم بسیار سریع بود. ولی اغلب به جای یافتن بهینه سراسری، نابلغ به بهینه محلی همگرا می شد. الگوریتم ژنتیک با احتمال کمتری به بهینه محلی همگرا می شد، ولی نشان داده شد بهبود آن به سمت بهینه سراسری کند است. با این وجود، هنگامی که دو الگوریتم با هم همکاری می کنند، ضعف های عملکرد جداگانه آنها مشاهده نمی شود. موفقیت ژنتیک پرندگان به دلیل قابلیت آن در تحقق بهترین ویژگی های هر الگوریتم تشکیل دهنده آن می باشد. نتایج نشان می دهد که مولفه های الگوریتم ژنتیک پرندگان با هم تعامل داشته تا ضعف های یکدیگر را برطرف کنند. این تعامل منجر به نتیجه بهترین نسبت به اجرای جداگانه الگوریتم های تشکیل دهنده شد. راهکارهای متعددی برای تحقیقات آینده وجود دارد. به منظور مشخص کردن رفتار جزئی الگوریتم، تحلیل کارایی از نظر میانه و انحراف معیار می تواند مفید واقع شود. یک امکان برای تغییر الگوریتم می تواند این باشد که فازهای الگوریتم ایستا نبوده و پویا باشند. تغییر بین فازها را می توان از طریق عدم پیشرفت کافی در فاز کنونی کنترل نمود. اگرچه مزیت فازهای جداگانه نیز نشان داده شد، مقایسه ای با الگوریتمی دوگانه ممکن است مفید باشد. در آخر، هرفاز جداگانه را می توان تنظیم نموده تا کارایی را بهبود بخشید.

Archive of SID



منابع

1. Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.
2. Engelbrecht, A. P. (2007). *Computational Intelligence*. West Sussex, England: John Wiley & Sons, Ltd.
3. Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. *IEEE*.
4. Eberhart, R., & Shi, Y. (1998). Comparison between genetic algorithms and particle swarm optimization. *Evolutionary Programming VII: Proc. 7th Ann. Conf. on Evolutionary Programming Conf.* Berlin: Springer-Verlag.
5. Castelli, M., Manzoni, L., & Vanneschi, L. (2011). The effect of Selection from Old Populations in Genetic Algorithms. *GECCO '11 Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*.
6. Juang, C.-F. (2004). A hybrid of Genetic Algorithm and Particls Swarm Optimization for Reccurent Network Design. *IEEE Transactions on Systems, Man, and Cybernetics*, 34(2).
7. Molga, M., & Smutnicki, C. (2005). Test functions for optimization needs.
8. Hedar, A.-R. (n.d.). *Global Optimization Test Problems*. Retrieved 10 16, 2011, from http://www-optimia.amp.i.kyotou.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm
9. Bersini, H., Dorigo, M., Langerman, S., Geront, G., & Gambardella, L. (1996, May 20-22). Results of the First International Contest on Evolutionary Optimisation (1st ICEO). *Proceedings of IEEE International Conference on Evolutionary Computation*, 611-615.
10. Brooks, J. (2012). *The Genetic Flock Algorithm*. Master's Thesis, Christopher Newport University

SID



سرویس های ویژه



سرویس ترجمه تخصصی



کارگاه های آموزشی



بلاگ مرکز اطلاعات علمی



عضویت در خبرنامه



فیلم های آموزشی

کارگاه های آموزشی مرکز اطلاعات علمی جهاد دانشگاهی



PROPOSAL
پروپوزال

پروپوزال نویسی و پایان نامه نویسی

دکتره تهرانی

کارگاه آنلاین
پروپوزال نویسی و پایان نامه نویسی



روش تحقیق و مقاله نویسی علوم انسانی

دکتره تهرانی

کارگاه آنلاین
روش تحقیق و مقاله نویسی علوم انسانی



ISI
Scopus

آشنایی با پایگاه های اطلاعات علمی بین المللی و ترند های جستجو

دکتره تهرانی

کارگاه آنلاین آشنایی با پایگاه های اطلاعات علمی بین المللی و ترند های جستجو