2nd INTERNATIONAL CONFERENCE ON
RESEARCH IN SCIENCE AND TECHNOLOGY
Istanbul-Turkey 14 March 2016
RSTCONF
دومین کنفرانس بین المللی
پژوهش در علوم و تکنولوژی
ترکیه - استانبول ۲۴ اسفند ۱۳۹۴

الگوریتم ژنتیک گروهی مبتنی بر الگوریتم پرندگان

محمد علی چاووشیان
آموزشکده فنی و حرفه ای ملاصدرا – 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 هردو را شرح داده و جزئیات الگوریتم را بیان می کند. هرسه الگوریتم با استفاده از هشت مسئله بهینه سازی استاندارد ادبیات موضوع با هم مقایسه می شوند. نشان داده می شود که الگوریتم ژنتیک پرندگان، کارآیی برتری در ۷۵٪ حالات تست شده دارد. در ۲۵٪ بقیه حالات، کارآیی 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. هرسه 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 اصولا 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 برازنده ترین افراد و جهش و تقاطع آنها تولید می شوند. این فرآیند تا زمانی که یکی از کروموزوم ها به برازندگی 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].

اخیرا، 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. هرذره، راه حلی ممکن را در فضای جستجو نشان می دهد و مکان هرذره برطبق 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، هرسه 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 GA وPSO

۲-۱-۱ شباهت ها

ارتباط واضحی بین GA و PSO وجود دارد که ترکیب این دو الگوریتم را ممکن می سازد. آنها هردو شامل دسته هایی از راه حل های مسئله موردنظر هستند. معمولا کروموزوم ها رشته های بیتی هستند. ولی ممکن است آرایه هایی از اعداد حقیقی نیز باشند. از طرف دیگر، یک ذره در PSO در یک فضای چندبعدی مکانی دارد که معمولا مجموعه ای از اعداد حقیقی است. با نوعی تغییر، می توان مکان را با مجموعه ای از بیت ها مشخص کرد [۲]. در این مقاله، هم کروموزوم ها و مکان های ذرات را آرایه هایی از اعداد حقیقی در نظر می گیریم. کروموزوم ها متناظر با مکان های ذرات هستند.

از آنجا که کروموزوم ها در هرنسل تغییر می کنند، این نسل ها متناظر با گام های زمانی گسسته هستند که در آنها، مکان های ذرات تغییر می کنند.

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

۲-۱-۲ تفاوت ها

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

بدیهی است که GA تصادفی تر از PSO می باشد. اگر والدین را با فرزندان شناسایی کنیم، ممکن است کروموزوم لزوما 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، الگوریتم های ژوتیک معمولا از نخبه گرایی استفاده می کنند. در این روش باید چند فردی که در نسلی بهترین هستند، تا نسل بعدی بدون تغییر باقی بمانندPSO .ها نیازی به نخبه گرایی ندارند؛ چرا که بهترین های قبلی هیچگاه از دست داده نمی شوند.

به دلایلی، PSO مشابه با نسخه های پیشرفته تر تپه نوردی هستند. مقادیری که قبل پیدا شده اند، به شدت جستجو را تحت تاثیر قرار می دهند. در GAها، مقادیر قبلی تنها بنیانی هستند که تغییرات از آنها نشأت می گیرند. آنها ذات تغییرات را تحت تاثیر قرار نمی دهند. اگرچه ممکن است تقاطع در GAها منجر به ترکیب راه حل های جزئی شود، ولی گاها آن ها را به هم می ریزد.

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

۲-۱-۳ - 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 هرفاز قابل تنظیم است، ولی به طور پیش فرض هریک 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 افزاینده بوده و از یک توزیع گاوسی با انحراف معیار کسری ثابت (معمولا ۰,۰۱) از بهینه سازی انتخاب می شود. نرخ جهش پیشفرض ۰,۱۰ است.

در فازPSO ، 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. هربار 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 جستجو طی 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 Juang [6] 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 وجهش 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 کارآیی سه الگوریتم مورد استفاده قرار گرفت. این مسائل از مسائلی ساده گرفته تا پیچیده انتخاب شده اند تا به ت صویری شفاف از چگونگی عملکرد هرالگوریتم دست یافته شود. این مسائل، مسائل ارزیابی الگوریتم معیار هستند که با هدف تست کارآیی الگوریتم های بهینه سازی طراحی شده اند [۷، ۸، ۹].

مسائل زیر مورد استفاده قرار گرفته اند:

1. Spherical $f(\vec{x})= \sum_{j=1}^{n} x_j^2$ where $-100 \le x_j \le 100$ -۱

2.Ackley $f(x)= -20 \exp\left(-.02\sqrt{\frac{1}{n}\sum_{j=1}^{n} x_j^2}\right) - \exp\left(\frac{1}{n}\sum_{j=1}^{n}\cos 2\pi x_i\right)+20+e$ where $-30 \le x_j \le 30$

3.Griewangk $f(\bar{X}) = - \sum_{j=1}^{n}\frac{x_j^2}{4000} - \prod_{j=1}^{n}\cos\left(\frac{x_j}{\sqrt{j}}\right) + 1$ where $-500 \le X_j \le 500$

4. Michalewicz $f(\bar{X}) = - \sum_{j=1}^{n}\sin(X_j)\left(\sin\left(\frac{jx_j^2}{\pi}\right)\right)^{2m}$ where m= 10 , $0 \le X_j \le \pi$

5. Rastrigin $f(\bar{X}) = 10n +\sum_{j=1}^{n}(x_j^2 - 10\cos(2\pi x_j))$ where $-5.12 \le x_j \le 5.12$

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.S chwefel $f(\bar{X}) = 418.9829\, n - \sum_{j=1}^{n} x_j \sin(\sqrt{|x_j|})$ where $-500 \leq x_j \leq 500$

8. Shekel $f(\bar{X}) = \sum_{j=1}^{10}[\sum_{j=1}^{4}(x_i - c_{ij})2 + \beta_j]$ 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 هرحالت، مقدار n=10 در روابط فوق مورد استفاده قرار گرفت. اندازه جمعیت براساس آزمایش و خطا ۵۶ انتخاب شد. به دلیل اینکه n=10، هرالگوریتم به ازای هرمسئله ۵۰ بار اجرا شد. هراجرای الگوریتم برای هرمسئله شامل ۵۰۰۰۰ گام بود. بعد از هراجرا، بهترین کارآیی الگوریتم ذخیره شد. سپس به ازای هرالگوریتم، میانگین این مقادیر در تمام اجراها درنظر گرفته شد. این روش تحلیل متوسط کارآیی هرالگوریتم را براساس هرمسئله ممکن می سازد.

۱-۳- رفتار پویای معمول

اجراهای که رفتار زمانی معمولی نشان دادند در ادامه آورده شده اند. به دلیل محدودیت در فضا، تنها خلاصه ای از نتایج در هرحالت ارائه شده است. برای جزئیات بیشتر به Brooks [۱۰] مراجعه نمایید.

برای مسئله کروی، الگوریتم بهینه سازی ذرات شروع خوبی داشته و در جهتی حرکت می کند که دو الگوریتم دیگر موفق به دستیابی به آن نشدند. اکثر بهبودهای پیشنهادی در الگوریتم ژنتیک پرندگان، هنگامی که در حالت بهینه سازی ذرات عمل می کرد، 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 بهینه سازی ذرات نبودند.

در مسئله ackley، الگوریتم بهینه سازی ذرات شروعی سریع داشت، ولی به زودی در بهینه محلی گیر کرده و بهبود دیگری نداشت. این امر در صورتی اتفاق می افتد که کل جمعیت به بهینه ای محلی محدود شوند و بهترین مکان ها (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 بهینه سازی ذرات رسیدند و پرندگان کمترین نتیجه را به دست آورد.

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

الگوریتم بهینه سازی ذرات در مسئله Michalewicz، در مقایسه با پرندگان و ژنتیک ضعیف عمل می کند. درحالیکه الگوریتم بهینه سازی ذرات در طول زمان بهبود می یابد، دو الگوریتم دیگر پیشرفت بسیار سریعتری دارند. پرندگان و ژنتیک نزدیک به هم پیشروی می کنند، ولی پرندگان جلوتر رفته و مقدار کمتری نسبت به ژنتیک پیدا می کند.

نتایج رابطه Rastrigin نیز مشابه با Michalewicz است. بهینه سازی ذرات در طول تکرارها بهبود می یابد؛ 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، بهینه سازی ذرات پیشرفت سریعی دارد. ژنتیک و پرندگان در طول تکرارهای متمادی مشابه هم عمل می کنند. در نهایت پرندگان می تواند به بهینه سازی ذرات رسیده و از آن جلو بزند. ژنتیک خط سیر بهبود یکنواختی دارد؛ ولی قادر به رسیدن به دو الگوریتم دیگر نیست.

در مسئلهSchwefel ، کارآیی بهینه سازی ذرات به شدت کمتری نسبت به دو الگوریتم دیگر نشان داد. برای درک دلیل این امر، اجراهای بیشتری با بازه های کوچکتری از مقادیر ممکن گرفته شد که تحت این اجراها بهینه سازی ذرات توانست به نسبت بهتر عمل کند. کارآیی هرسه الگوریتم در رابطه Shekel نزدیک به هم بود. بهینه سازی ذرات کندتر از همه شروع کرده و پرندگان هنگامی که از ژنتیک به بهینه سازی ذرات تغییر حالت می دهد، از همه جلو می افتد. پس از این، پرندگان به بهترین کارآیی دست می یابد. ژنتیک و بهینه سازی ذرات در اوایل با هم رقابت می کنند، ولی همانطور که تکرارهای بیشتری انجام می شود، ژنتیک می تواند به کارآیی بهتری نسبت به بهینه سازی ذرات دست یابد.

۲-۳ - 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	*1.74 x 10^{-5}*	2.00 x 10^{-19}	0
Ackley	1.56 x 10^{-3}	**1.40 x 10^{-10}**	*6.67 x10^{-1}*
Griewangk	*5.45 x 10^{-2}*	**5.90 x 10^{-3}**	4.30 x10^{-2}
Michalewicz	4.89 x 10^{-7}	0	*4.06 x 10^{-1}*
Rastrigin	8.66 x 10^{-6}	**1.71 x 10^{-15}**	*4.29*
RosenBrock	*7.73 x 10^{-3}*	**9.04 x 10^{-15}**	3.66 x 10^{-3}
Schwefel	5.38 x 10^{-3}	**1.27 x 10^{-4}**	*1.16 x10^{3}*
Shekel	**2.21 x 10^{-1}**	**2.21 x 10^{-1}**	*2.23 x10^{-1}*

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 کارآیی را دارد. در نهایت، این الگوریتم دارای بالاترین کارآیی ۷۵٪ اندازه گیری ها بود. در ۲۵٪ باقیمانده، پرندگان از یک الگوریتم دیگر کارآیی بالاترین نشان داد. پرندگان تنها الگوریتمی بود که در هیچ اجرایی بدترین کارآیی را نداشت. مطابق با ویژگی های معمول بهینه سازی ذرات، این الگوریتم بسیار سریع بود. ولی اغلب به جای یافتن بهینه سراسری، نابالغ 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 هرالگوریتم 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 کارآیی 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 کارآیی را بهبود بخشید.

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 '11Proceedings 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 Reccurrent 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-optima.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.