به نام خدا

دانشگاه آزاد اسلامی
واحد قزوین

آشنایی با الگوریتم های تکاملی

استاد : دکتر رضوی

گرد آورندگان

حسن دادگر
مهرداد جوکار
وحید خورانی

پاییز 87

الگوریتم‌های ژنتیک

الگوریتم‌های ژنتیک، پراستفاده‌ترین و شناخته‌شده‌ترین الگوریتم‌های تکاملی هستند. اجزای زیر ساختی فرایند تکاملی، تجدید نسل، جهش، رقابت و انتخاب هستند. تجدید نسل، فرایندی است که در طی آن یک عنصر جمعیت برنامه‌ی ژنی خود را به فرزندان خود انتقال می‌دهد. انتقال برنامه‌ی ژنی در معرض خطایی قرار دارد که همان جهش را تشکیل می‌دهد. جهش می‌تواند هم موجب بهبود در وضعیت فرزند شود و هم اینکه آن را خراب کند. محدودیت منابع موجود در محیط نیز موجب ایجاد رقابت می‌شود. قواعد اساسی الگوریتم ژنتیک برای اولین بار در سال *1962* توسط هالند معرفی گردید و تا به امروز کاربردهای فراوانی در بهینه سازی توابع و شناسایی سیستم پیدا کرده‌اند. الگوریتم ژنتیک، یک تکنیک جستجوی فراگیر است که اساس آن روی عمل ژنتیک طبیعی و تئوری انتخاب طبیعی داروین بوده و در واقع تبادل اطلاعات ساخت‌یافته، به طور تصادفی است. قانون انتخاب طبیعی، بدین صورت است که تنها گونه‌هایی از یک جمعیت ادامه نسل می‌دهند که بهترین خصوصیات را داشته باشند و آنهایی که این خصوصیات را نداشته باشند، به تدریج و در طی زمان از بین می‌روند. بدین ترتیب می‌توان دید که طبیعت با بهره‌گیری از یک روش بسیار ساده (حذف تدریجی گونه‌های نامناسب و در عین حال تکثیر بالاتر گونه‌های بهینه)، توانسته است دائما هر نسل را از لحاظ خصوصیات مختلف ارتقا بخشد. در واقع می‌توان تکامل طبیعی را به این‌صورت خلاصه کرد: جست‌وجوی کورکورانه (تصادف یا *Blind Search*) + بقای قوی‌تر.

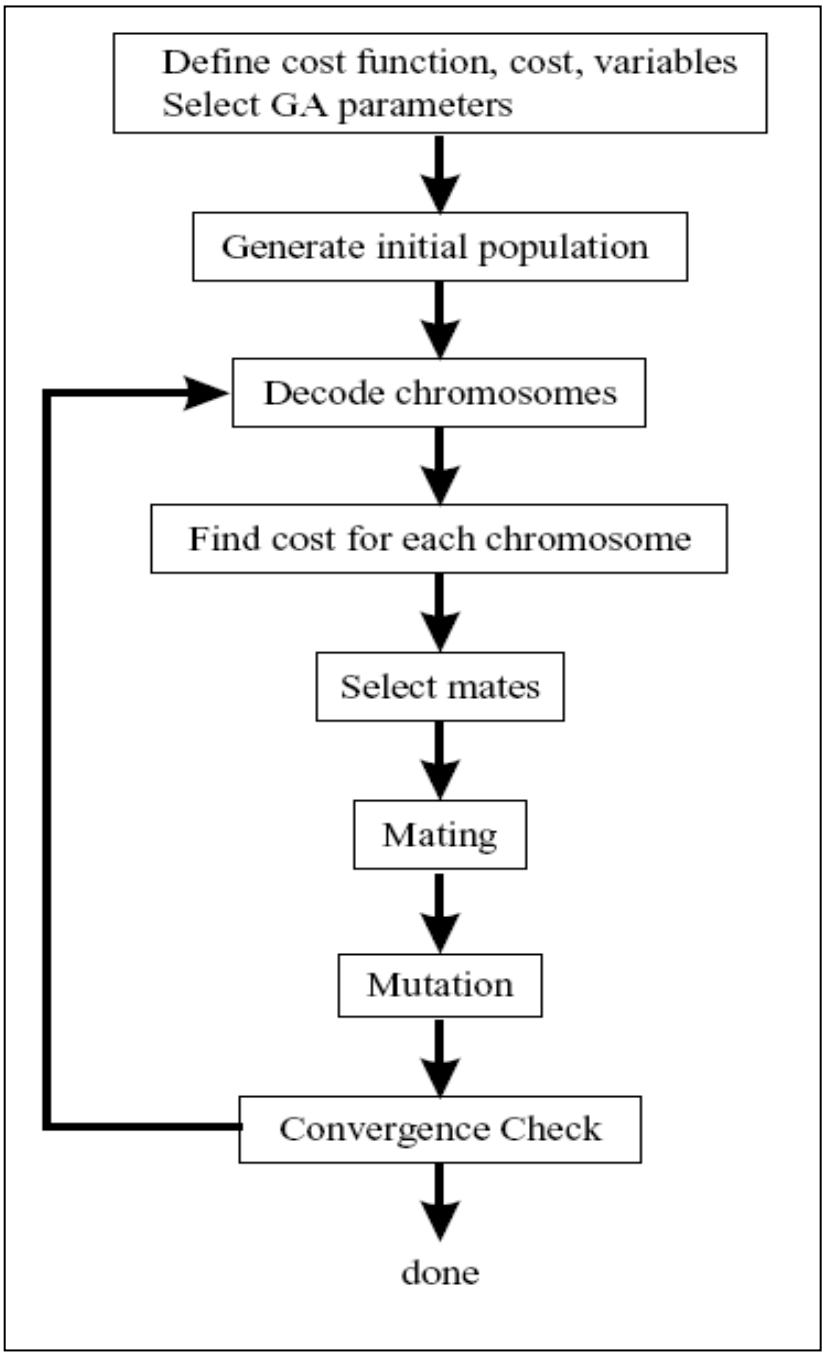
اصول بنیادین الگوریتم ژنتیک نیز از ژنتیک طبیعی و فرضیه سیر تکاملی نشات گرفته و از جمعیتی، شامل رشته‌های عددی و عموما دودویی استفاده می‌کنند. این رشته‌های عددی که بیانگر پارامترهای سیستم، می‌باشند در راستای بیشینه (یا کمینه) نمودن یک تابع، تعیین می‌شوند. این تابع که قابلیت توصیف سیستم توسط آن رشته عددی را بیان می‌دارد، تابع شایستگی[1] یا تابع ارزش[2] و یا تابع هزینه[3] خوانده می‌شود و عموما مقدار نرمالیزه آن استفاده می‌گردد و می‌تواند هر تابع غیر خطی، مشتق ناپذیر، گسسته و حتی یک الگوریتم باشد. همانگونه که قبلاً نیز بیان شد، الگوریتم ژنتیک، یک روش جستجوی مطلق می‌باشد و بنابراین تفاوت عمده‌ای با روش‌های کلاسیک جستجوی محلی دارد. در جدول *1*، مهم‌ترین تفاوت‌های الگوریتم ژنتیک، با روش‌های عادی بهینه‌سازی ذکر شده است.

جدول *1*. تفاوتهای الگوریتم ژنتیک با روشهای استاندارد بهینه سازی

الگوریتم ژنتیک	الگوریتم‌های استاندارد
در هر تکرار، یک جمعیت تولید می‌کند که نهایتا جمعیت به یک راه حل بهینه می‌رسد.	در هر تکرار یک نقطه تولید می کنند که نهایتا دنباله نقطه‌ها به یک راه حل بهینه می‌رسند.
هر جمعیت بعدی را بر اساس محاسباتی که گزینه‌های تصادفی می‌یابند، پیدا می‌کنند.	هر نقطه بعدی را بر اساس محاسبه معین پیدا می‌کنند. (ممکن است فقط در مینیمم محلی نوسان کند.)

شکل *2* شمای کلی الگوریتم ژنتیک را نشان می‌دهد. در ادامه اجزای اصلی این الگوریتم به طور خلاصه، مورد بررسی قرار می‌گیرند.

[1] Fitness Function
[2] Value Function
[3] Cost Function



شکل 2: شمای کلی الگوریتم ژنتیک

عملگرها و عناصر الگوریتم ژنتیک به طور خلاصه عبارتند از:

1-1-1- کروموزوم

هر کروموزوم به منزله یک جواب مسأله است. در الگوریتم ژنتیک دودویی، کروموزوم به صورت ترکیبی از صفر و یک نمایش داده می‌شود. معمولا جمعیت اولیه کروموزوم‌ها به صورت تصادفی و در فضایی با توزیع احتمال یکسان، تولید می‌شوند.

1-1-2- تابع هدف

به تابعی گفته میشود که می‌خواهیم بیشینه و کمینه آن را پیدا نماییم. هر کروموزوم، جزو دامنه این تابع، میباشد.

1-1-3- تکثیر

اساس این عملگر، اصل بقا و شایستگی در جمعیت می‌باشد. ابتدا، یک زیر جمعیت اولیه، به طور تصادفی و بدون جایگزینی از کل جامعه انتخاب می‌گردد. از میان بهترین این جعیت، تعدادی به عنوان والد انتخاب شده و فرزندان از روی آن‌ها تولید می‌شوند.

1-1-4- تابع ارزش

با توجه به این تابع، به هر کروموزوم عددی نسبت داده می‌شود که بیانگر ارزش آن کروموزوم است و در هر نسل بزرگ بودن آن بیانگر میزان نزدیک بودن هر کروموزوم به جواب نهایی می‌باشد.

1-1-5- انتخاب

انتهای هر نسل جمعیتی از کروموزوم‌ها به نسل بعد منتقل می‌شوند تا کروموزوم‌ها، از آنها تولید شوند.

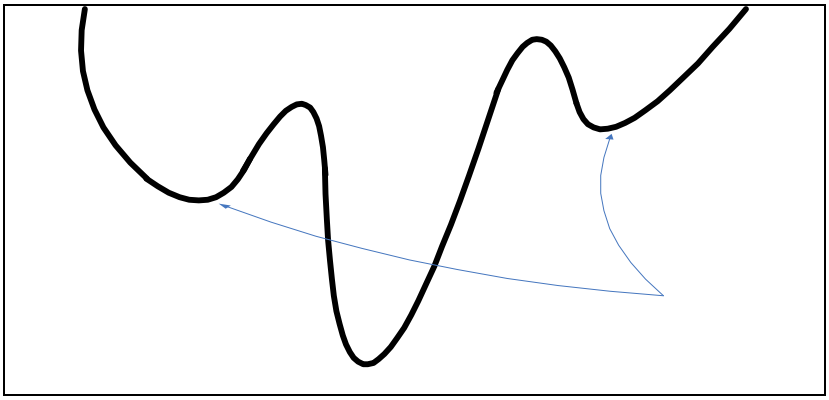
1-1-6- جهش

این عملگر دقیقا همانند جهش کروموزومی در طبیعت عمل می‌کند و با احتمال خیلی کم، و در سطح بیت، تغییراتی به وجود می‌آورد. با این روش، احتمال گیر افتادن در کمینه‌های محلی کاهش می‌یابد.

آبکاری فلزات شبیه سازی شده

در اوایل دهه *1980*، روش آبکاری فلزات شبیه‌سازی شده[4] *(SA)*، توسط *Kirkpatrick* و همکارانش (*1983*)، معرفی شد. این الگوریتم بر پایه‌ی نظریاتی بود که در اوایل دهه‌ی *1950*، فرمولبندی شده بود. این روش، فرایند آب‌کاری را که در آن، یک ماده تا دمای بالاتر از نقطه‌ی ذوبش گرما داده شده و سپس به آرامی سرد می‌شود تا یک شبکه کریستالی را ایجاد کند؛ شبیه‌سازی می‌کند. ایجاد شبکه کریستالی به گونه‌ای است که توزیع احتمالاتی انرژی را کمینه می‌کند. این شبکه‌ی کریستالی که از میلیون‌ها اتم به طور کامل جهت‌دهی‌شده تشکیل یافته است، مثال زیبایی از یافتن ساختار بهینه توسط طبیعت است.

با دادن انرژی و سرد کردن، ماده، در یکی از نقاط تعادل خود قرار می‌گیرد. اگر عمل سرد کردن با مهارت، انجام پذیرفته باشد، می‌توان، بافت مولکولی را از میان چندین حالت تعادل مختلف، در حالتی قرار داد که مطلوب ماست (حالت بهینه). شکل *3* مینیمم‌های محلی و ساختار مطلوب (مینیمم گلوبال) را نشان می‌دهد.



[4] simulated annealing

شکل 3: مینیمم‌های محلی و ساختار مطلوب (مینیمم گلوبال)

روش کار الگوریتم SA به این صورت است که ابتدا از یک نقطه‌ی تصادفی شروع می‌کنیم. اگر گام بعدی در جهت اهداف ما یعنی کمینه کردن تابع هزینه باشد، آن را می‌پذیریم و اگر در جهت افزایش باشد، به نفع اهداف غایی، **با** *احتمالی*، این گام را می‌پذیریم. شکل 4، شبه‌کد مربوط به الگوریتم آب‌کاری فلزات شبیه‌سازی شده را نشان می‌دهد.

1. $\boldsymbol{x}$ را به تصادف انتخاب می‌کنیم.
2. $\boldsymbol{P}$ (جهت حرکت اولیه) و η ضریب گام نیز به تصادف انتخاب می‌شود.
3. $\boldsymbol{x}^{+} = \boldsymbol{x}^{-} - \eta \boldsymbol{p}^{-}$
4. اگر $f(\boldsymbol{x}^{+}) < f(\boldsymbol{x}^{-})$ حرکت پذیرفته می‌شود.
5. اگر $f(\boldsymbol{x}^{+}) > f(\boldsymbol{x}^{-})$ حرکت با احتمال $p(\Delta f)$ پذیرفته می‌شود. که

$$\Delta f = f(\boldsymbol{x}^{+}) - f(\boldsymbol{x}^{-})$$

$$p(\Delta f) = \frac{1}{1 + e^{-\Delta f / T(t)}}$$

6. بازگشت به گام 2 و تکرار الگوریتم تا برآورده شدن یک شرط توقف

شکل 4: شبه‌کد مربوط به الگوریتم آب‌کاری فلزات شبیه‌سازی شده

$T(t)$ را معمولاً، به شکل زیر در نظر می‌گیریم.

$$T(t) = \frac{T_0}{1 + \ln(t)}, \qquad t = 1, 2, 3, \ldots$$

با افزایش $T(t)$، تعداد حرکت‌های سربالایی را کاهش می‌دهیم. یعنی یک انرژی به سیستم می‌دهیم (بالا رفتن). در سال 1983 ثابت شد که اگر $T(t)$ دارای فرم بالا و یا کندتر از آن باشد (با این شرط که حتماً کاهشی بوده و به صفر میل کند)؛ در حالت تکرار بی‌نهایت، الگوریتم حتماً همگرا می‌شود.

الگوریتم رقابت استعماری

در این فصل، الگوریتم جدیدی برای جستجوی عام معرفی می‌شود که از رقابت‌های استعماری الهام گرفته شده است. این الگوریتم، از چندین کشور در حالت اولیه شروع می‌شود. همه‌ی کشورها، به دو دسته تقسیم می‌شوند: امپریالیست و مستعمره. رقابت امپریالیستی، هسته‌ی اصلی این الگوریتم را تشکیل می‌دهد و باعث می‌شود که کلیه‌ی کشورها به مینیمم مطلق تابع هزینه، همگرا شوند

شکل 1 فلوچارت الگوریتم پیشنهادی را نشان می‌دهد. همانند دیگر الگوریتم‌های تکاملی، این الگوریتم، نیز با تعدادی جمعیت اولیه تصادفی که هر کدام از آنها یک "کشور" نامیده می‌شوند؛ شروع می‌شود. تعدادی از بهترین عناصر جمعیت (معادل نخبه‌ها در الگوریتم ژنتیک) به عنوان امپریالیست[5] انتخاب می‌شوند. باقیمانده جمعیت نیز به عنوان مستعمره[6]، در نظر گرفته می‌شوند. استعمارگران بسته به قدرتشان، این مستعمرات را با یک روند خاص که در ادامه می‌آید؛ به سمت خود می‌کشند. قدرت کل هر امپراطوری، به هر دو بخش تشکیل دهنده آن یعنی کشور امپریالیست و مستعمرات آن، بستگی دارد. در حالت ریاضی، این وابستگی با تعریف قدرت امپراطوری به صورت مجوع قدرت کشور امپریالیست، به اضافه در صدی از میانگین قدرت مستعمرات آن، مدل شده است.

با شکل‌گیری امپراطوری‌های اولیه، رقابت امپریالیستی میان آن‌ها شروع می‌شود. هر امپراطوری‌ای که نتواند در رقابت استعماری، موفق عمل کرده و بر قدرت خود بیفزاید (و یا حداقل از کاهش نفوذش جلوگیری کند)، از صحنه رقابت استعماری، حذف خواهد شد. بنابراین بقای یک امپراطوری، وابسته به قدرت آن در جذب مستعمرات امپراطوری‌های رقیب، و به سیطره در آوردن آنها خواهد بود. در نتیجه، در جریان رقابت‌های امپریالیستی، به تدریج بر قدرت امپراطوری‌های بزرگتر افزوده شده و امپراطوری‌های ضعیف‌تر، حذف خواهند شد. امپراطوری‌ها برای افزایش قدرت خود، مجبور خواهند شد تا مستعمرات خود را نیز پیشرفت دهند.

بدین ترتیب، با گذشت زمان، مستعمرات، از لحاظ قدرت به امپراطوری‌ها نزدیک‌تر خواهند شد و شاهد یک نوع همگرایی خواهیم بود. حد نهایی (زمان توقف الگوریتم)، زمانی است که یک امپراطوری واحد در دنیا داشته باشیم، با مستمراتی که از لحاظ موقعیت، به خود کشور امپریالیست، خیلی نزدیک هستند

1-چند نقطه تصادفی روی تابع انتخاب کرده و امپراطوری‌های اولیه را تشکیل بده.

2-مستعمرات را به سمت کشور امپریالیست حرکت بده.

3-اگر مستعمره‌ای در یک امپراطوری، وجود داشته باشد که هزینه‌ای کمتر از امپریالیست داشته باشد؛ جای مستعمره و امپریالیست را با هم عوض کن.

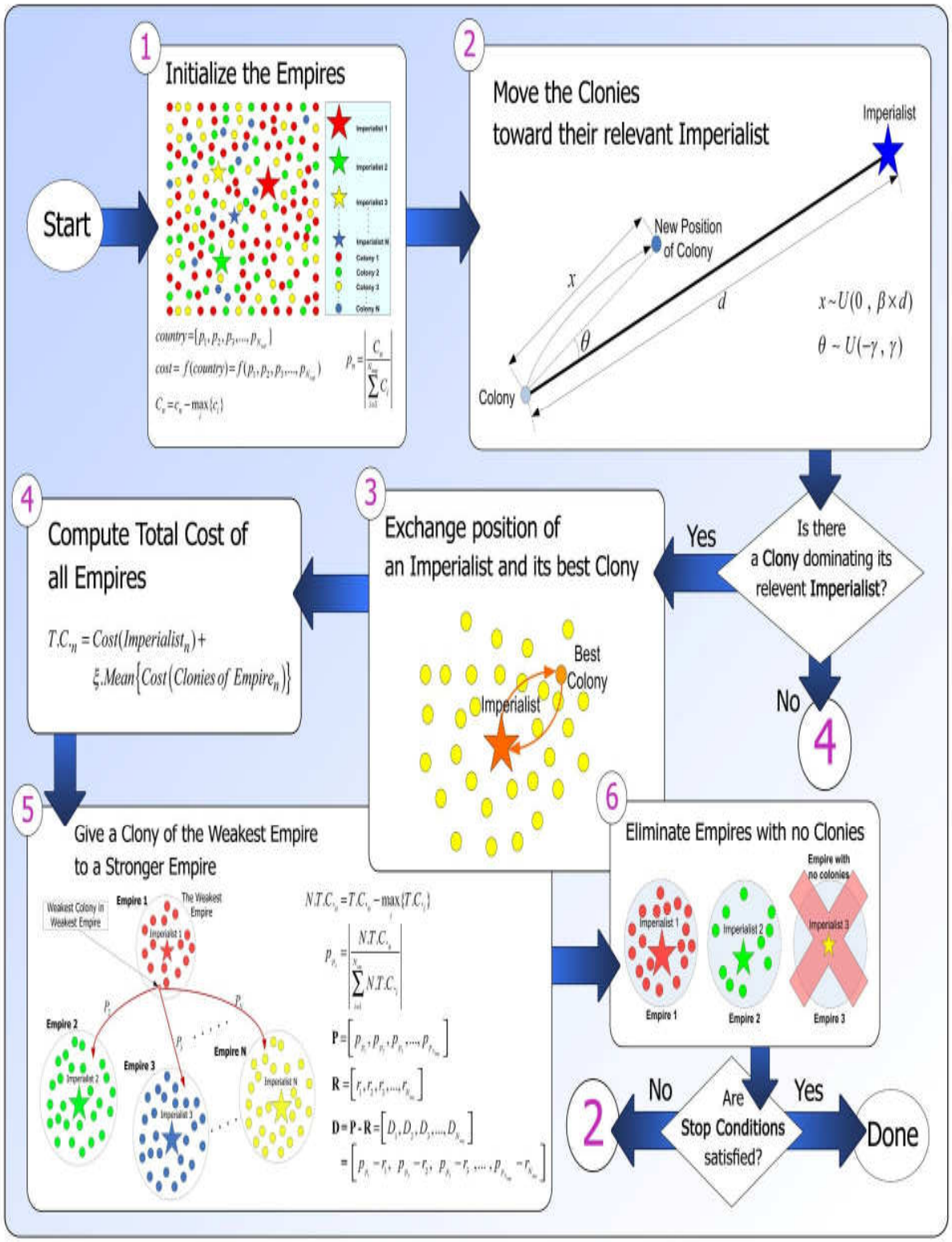
4-هزینه‌ی کل یک امپراطوری را حساب کن (با در نظر گرفتن هزینه‌ی امپریالیست و مستعمراتشان).

5-یک مستعمره از ضعیف‌ترین امپراطوری انتخاب کرده و آن را به امپراطوری‌ای که بیشترین احتمال تصاحب را دارد، بده.

6-امپراطوری‌های ضعیف را حذف کن.

7-اگر تنها یک امپراطوری باقی مانده باشد، توقف کن وگرنه به 2 برو.

[5] Imperialist
[6] Colony



Dadgar24@yahoo.com

Colonial competitive optimization algorithm

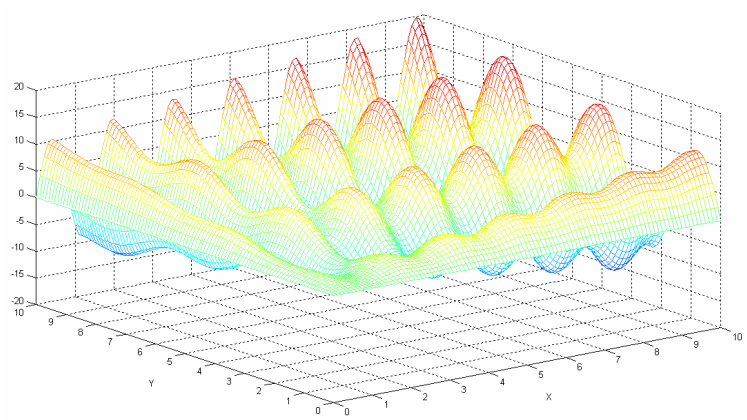
Country $=[x\ ,\ y]$

Cost Function

$$f = x.\sin(4x) + 1.1y.\sin(2y)$$

$$0 < x, y > 10$$

$$minimum:\ f(9.039, 8.668) = -18.5547$$



تابع بالا یکی از توابعی است که دارای مینیمم محلی زیادی است و فقط یک مینیمم مطلق دارد و معیاری است برای سنجش الگوریتم های تکاملی ابداع شده است و به تابع راسل معروف می باشد

تشکیل کشورها و استخراج امپریالیست ها از مهمترین گامهای حل این مساله می باشد که در ذیل توضیح داده می شود

استاد : دکتر فرزاد رضوی ارایه دهنده گان :1- حسن دادگر 2-مهرداد جوکار 3- وحید خورانی
Dadgar24@yahoo.com

	x	y	$cost$
Country1 =	[8.1472	1.5761]	[7.4926]
Country2 =	[9.0579	9.7059]	[-3.3176], I_3
Country3 =	[1.2699	9.5717]	[1.8636]
Country4 =	[9.1338	4.8538]	[-9.8782], I_1
Country5 =	[6.3236	8.0028]	[-1.5641]
Country6 =	[0.9754	1.4189]	[-0.2050]
Country7 =	[2.7850	4.2176]	[1.1215]
Country8 =	[5.4688	9.1574]	[-4.5020], I_2
Country9 =	[9.5751	7.9221]	[4.2326]
Country10 =	[9.6489	9.5949]	[11.059]

مرحله 3

در این مرحله تعویض امپریالیست ها دار
در مجموعه امپراطوری ها اتفاق می افت

[7.1472	3.4500]	[-0.0165]
[9.0579	9.7059]	[-3.3176], I_3
[8.9090	9.2926]	[-10.521], I_1
[9.1338	4.8538]	[-9.8782]
[7.3210	6.6540]	[-1.2542]
[4.2345	3.4210]	[-1.9959]
[6.9300	2.4356]	[1.0023]
[5.4688	9.1574]	[-4.5020], I_2
[3.4213	9.3000]	[0.5512]
[4.7893	4.9000]	[-0.5251]

Dadgar24@yahoo.com

مرحله 4

$$T.C._1 = -10.873$$
$$T.C._2 = -4.7327$$
$$T.C._3 = -3.3386$$

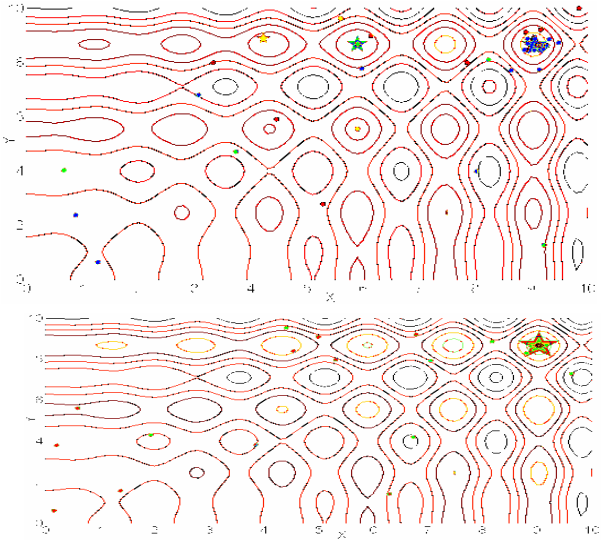
برای رسیدن به امپراطوری واحد و حذف ضعیف ترین امپراطور صورت می امپراطوری ضعیف تر صورت می گیرد

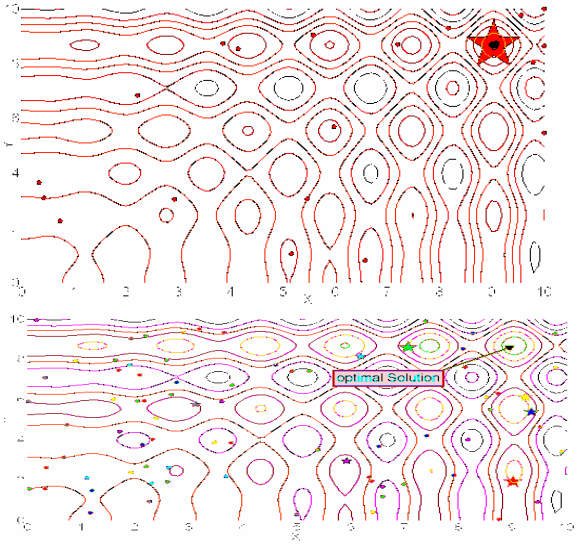
$$\boldsymbol{P} = [0.8495 \;\; 0.1505 \;\; 0]$$
$$\boldsymbol{R} = [0.4521 \;\; 0.2345 \;\; 0.1487]$$
$$\boldsymbol{D} = [\mathbf{0.3974} \;\; -0.084 \;\; -0.1487]$$

$$N.T.C._1 = -7.5344$$
$$N.T.C._2 = -1.3349$$
$$N.T.C._3 = 0$$

$$P_{p1} = 0.8495$$
$$P_{p1} = 0.1505$$
$$P_{p1} = 0$$

شمایل گرافیکی الگوریتم رقابت استعماری در حال اجرا





Dadgar24@yahoo.com

سپاسگذاری و تشکر:

بر خودمان وظیفه می دانیم از زحمات دکتر کارو لوکس و مهندس اتشپز بخاطر راهنمایی ارزشمندشان در معرفی الگوریتم رقابت استعماری قدردانی نماییم.

References:

-Atashpaz-Gargari E. and Lucas C., "Colonial Competitive Algorithm: 2007 IEEE Congress on Evolutionary Computation (CEC 2007)

-Davis, L., 1991. Handbook of Genetic Algorithms. Van Nostrand, New York.

-Goldberg, D.E., 1989. Genetic Algorithm in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

-Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220, 671–680.

استاد : دکتر فرزاد رضوی ارایه دهنده گان :1- حسن دادگر 2-مهرداد جوکار 3- وحید خورانی
Dadgar24@yahoo.com