***چشم انداز مدیریت صنعتی***
شماره ۲- تابستان ۱۳۹۰
ص‌ص ۱۲۳- ۱۰۷

# طراحی یک الگوریتم فرا ابتکاری جدید بر اساس رفتار توابع ریاضی xCos(x) و tanh(x)

**سید هادی میرقادری *، مصطفی زندیه ****

**چکیده**

امروزه استفاده از روش‌های فرا ابتکاری برای دست‌یابی به پاسخ‌های رضایتبخش در بهینه‌یابی ترکیبیاتی رشد چشمگیری یافته است. به دلیل نزدیک شدن مسائل به شرایط موجود در دنیای واقعی و در نتیجه افزایش پیچیدگی مسائل و ناتوانی روش‌های ریاضی فعلی برای ارائه نقطه بهینه با صرف معقول منابع، این اقبال تشدید شده است. توسعه روش‌های فرا ابتکاری معمولاً با بررسی نحوه بهینه‌یابی در طبیعت و الهام گرفتن از آن صورت می‌گیرد که از جمله می‌توان به الگوریتم ژنتیک، الگوریتم مورچگان و شبیه‌سازی تبرید اشاره کرد.

الگوریتم پیشنهادی این مقاله، با بررسی رفتار جالب توجه دو تابع xCos(x) و tanh(x) در حلقه‌های تکرار، توسعه یافته است و روشی برای یافتن همسایگی در توابع پیوسته ارائه می‌دهد که نسبت به الگوریتم بهینه‌یابی شبیه‌سازی تبرید و الگوریتم شبیه‌سازی تبرید مبتنی بر تئوری ابر، عملکرد بهتری از نظر دقت و سرعت دارد. برتری الگوریتم پیشنهادی به دو الگوریتم یاد شده، با مقایسه عملکرد این الگوریتم‌ها در یافتن نقطه (نقاط) بهینه هفت تابع پیوسته معروف به اثبات رسید.

**کلید واژه‌ها: فرا ابتکاری، جستجوی همسایگی، بهینه‌یابی پیوسته، شبیه‌سازی تبرید.**

---

تاریخ دریافت مقاله: ۸۹/۱۰/۲، تاریخ پذیرش مقاله: ۹۰/۲/۲۵.

*دانشجوی دکتری مدیریت تولید و عملیات دانشگاه شهید بهشتی. (نویسنده مسئول).
E-mail: Hadi.Mirghaderi@yahoo.com

** استادیار دانشگاه شهید بهشتی.

**مقدمه**

یک الگوریتم، توصیفی از گام‌هایی است که به‌گونه‌ای مناسب‌تر در یک برنامه کامپیوتری پیاده‌سازی می‌شود تا تقریبی از یک نقطه بهینه یافت شود. طراحی یک الگوریتم می‌تواند چند هدف داشته باشد؛ از جمله دست‌یابی به نقطه بهینه محلی، دستیابی به نقطه بهینه سراسری، یافتن همه نقاط بهینه سراسری، یافتن همه نقاط بهینه سراسری و محلی [۷].

الگوریتم‌هایی که در زمینه بهینه‌یابی ترکیبیاتی[۱] ارائه شده‌اند، قابل دسته‌بندی به دو گروه الگوریتم‌های کامل[۲] و الگوریتم‌های تقریبی[۳] هستند. الگوریتم‌های کامل، متضمن یافتن یک جواب بهینه در زمان محدود برای همه مصادیق یک مسأله بهینه‌یابی در اندازه متناهی (محدود) هستند [۲].

الگوریتم‌های تقریب[۴] رسماً در دهه ۱۹۶۰ به منظور تولید جواب‌های نزدیک به بهینه[۵] معرفی شدند. این الگوریتم‌ها برای آن دسته از مسائل بهینه‌یابی به‌کار می‌روند که با روش‌های محاسباتی زمان خود نمی‌توان آنها را به صورت اثربخش حل کرد. با ظهور تئوری غیرچندجمله‌ای تآم[۶] در اوایل دهه ۱۹۷۰، این حوزه علمی مهم‌تر شد، زیرا نیاز به تولید جواب‌های نزدیک به بهینه برای مسائل بهینه‌یابی غیرچندجمله‌ای سخت[۷] به منظور غلبه بر یاغی‌گری محاسباتی[۸] این مسائل به شدت احساس می‌شد. الگوریتم‌های آن دهه، جواب نزدیک به بهینه در زمان کوتاه به یک مسأله می‌داد و برای سایر مسائل نیز به سختی جواب زیربهینه[۹] تولید می‌کرد [۶].

روش‌های تقریبی، عموماً مبتنی بر دو اصل پایه‌ای هستند: تکنیک‌های ابتکاری سازنده[۱۰] و روش‌های جستجوی محلی[۱۱]. تکنیک‌های ابتکاری سازنده، مربوط به فرایند ساختن جواب‌های اولیه قبل از انجام عملیات دیگری است. جستجوی محلی می‌تواند به عنوان ساز وکار ابتکاری در نظر گرفته شود که در آن، همسایه‌های جواب فعلی به عنوان جایگزین‌های بالقوه جواب فعلی بررسی می‌شوند. اگر یکی از همسایه‌های جواب فعلی پذیرفته شود، حرکت به سوی جواب جدید آغاز می‌شود و همسایه‌های این جواب مورد توجه قرار می‌گیرند [۴].

1. Combinatorial Optimization
2. Complete Algorithms
3. Approximate Algorithms
4. Approximation Algorithms
5. Near-Optimal Solutions
6. Theory of NP-Completeness
7. NP-hard Optimization Problem
8. Computational Intractability
9. Suboptimal Solutions
10. Constructive Heuristics
11. Local Search Methods

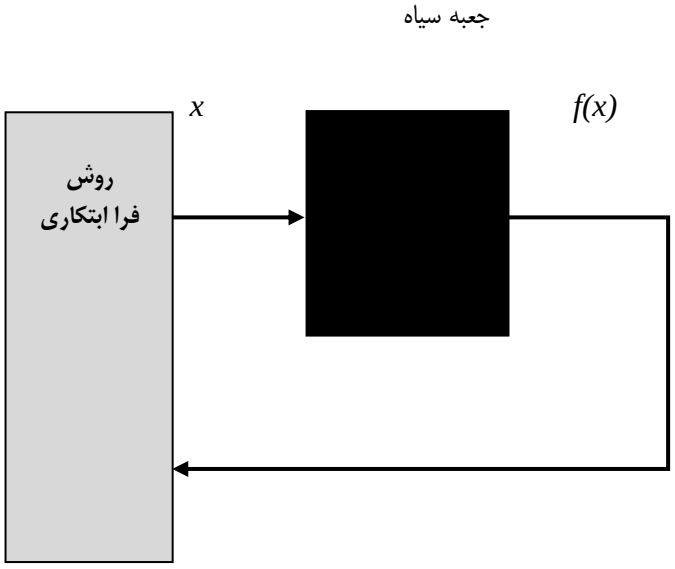
برخی منابع علمی، تکنیک‌های ابتکاری را به دو خانواده ابتکاری‌های خاص[1] و فرا ابتکاری‌ها[2] تقسیم کرده‌اند [۱۱]. ابتکاری‌های خاص برای حل یک مسأله خاص و یا یک مصداق از آن طراحی می‌شوند، در حالی که فرا ابتکاری‌ها الگوریتم‌های عموم‌منظوره[3] هستند که قابل استفاده در تقریباً همه مسائل بهینه‌یابی هستند.

## الگوریتم‌های فرا ابتکاری

در سی سال گذشته، نوع جدیدی از الگوریتم‌های تقریب ظهور یافته‌اند که اساساً هدف از آنها ترکیب روش‌های ابتکاری در چارچوبهای کلان‌تر به منظور کاوش کارا و اثربخش فضای جستجو می‌باشد. امروزه از این روش‌ها با عنوان روشهای فرا ابتکاری(متاهیوریستیک) نام برده می‌شود. این واژه را اولین بار «گلوور»[4] در ۱۹۸۶ به کار برد که از ترکیب دو واژه یونانی «متا» و «هیوریستیک» ساخته شده است. پیشوند «متا» به معنای فراتر یا در سطحی بالاتر است و «هیوریستیک» به معنای یافتن است. قبل از پذیرش عمومی واژه فرا ابتکاری، عبارت «روش ابتکاری نوین»[5] برای اینگونه روش‌ها به‌کار می‌رفت [۱]. یک روش فرا ابتکاری، در حقیقت، یک روش ابتکاری برای حل یک طبقه بسیار عمومی از مسائل است و ترکیبی کارآمد از توابع هدف یا روش‌های ابتکاری مبتنی بر توابع هدف می‌باشد [۱۳].

مزیت اصلی استفاده از روش‌های فرا ابتکاری، وجود مفروضات محدود در فرموله کردن مدل است، در حالی که این امر در برنامه‌ریزی ریاضی مصداق ندارد. برخی از مسائل بهینه‌یابی را نمی‌توان با نمادهای تحلیلی–ریاضی بدون ابهام فرموله کرد. در واقع، تابع هدف در این موارد، مانند یک جعبه سیاه[6] است. در بهینه‌یابی جعبه سیاه، هیچ قاعده تحلیلی برای هدف وجود ندارد [۱۱] (نمودار ۱).

1. Specific Heuristics
2. Metaheuristics
3. General-Purposer
4. Glover
5. Modern Heuristics
6. Black box




شکل ۱. سناریوی جعبه سیاه برای تابع هدف

الگوریتم‌های فرا ابتکاری بسیاری ارائه شده‌اند. این الگوریتم‌ها کارایی بالاتری برای برخی مسائل دارند و در برخی مسائل با مشکلاتی در زمینه نزدیک شدن به جواب بهینه مواجه می‌شوند. در حقیقت، با توجه به ساختار مسأله و نوع پیچیدگی آن، یک فرد خبره در زمینه الگوریتم‌های فرا ابتکاری، یک الگوریتم را برای حل برمی‌گزیند و پارامترهای مربوط به آن الگوریتم را با استفاده از روش علمی طراحی آزمایش‌ها[1] یا روش آزمون و خطا تنظیم می‌کند. جدول ۱ مهمترین روش‌های فرا ابتکاری را نشان می‌دهد.

1. Design of Experiments(DOE)

بس‌یاری از فرا ابتکاری‌ها ملهم از فرایندهای طبیعی یا فیزیکی هستند. بهینه‌یابی لانه مورچه[1]، الگوریتمهای تکاملی[2] و شبیه‌سازی تبرید[3]، مثالهایی از چنین الگوریتم‌هایی هستند. ACO و EAs از طبیعت و SA از فرایند سردسازی فلزات در تولید فلزات مستحکم الهام گرفته شده‌اند [۳].

*جدول۱. مهمترین روش‌های فرا ابتکاری*

| سال | ابداع‌کننده | الگوریتم |
|---|---|---|
| ۱۹۷۰ | Hastings | Metropolis-Hastings algorithm |
| ۱۹۷۵ | Holland | genetic algorithm |
| ۱۹۸۰ | Smith | genetic programming |
| ۱۹۸۳ | Kirkpatrick et al. | simulated annealing |
| ۱۹۸۶ | Glover | tabu search |
| ۱۹۸۶ | Farmer et al. | artificial immune system |
| ۱۹۸۹ | Moscato | memetic algorithm |
| ۱۹۹۲ | Dorigo | ant colony algorithm |
| ۱۹۹۳ | Fleming | MOGA for multiobjective optimization |
| ۱۹۹۴ | Fonseca | NSGA for multiobjective optimization |
| ۱۹۹۵ | Kennedy and Eberhart | particle swarm optimization |
| ۱۹۹۶ | Hansen and Ostermeier | CMA-ES |
| ۱۹۹۷ | differential evolution | Storn and Price |
| ۱۹۹۷ | cross entropy method | Rubinstein |
| ۲۰۰۱ | Geem et al. | harmony search |
| ۲۰۰۲ | Deb et al. | NSGA-II for multiobjective optimization |
| ۲۰۰۴ | Nakrani and Tovey | bee colony optimization |
| ۲۰۰۵ | Krishnanand and Ghose | Glowworm swarm optimization |
| ۲۰۰۵ | Karaboga | Artificial Bee Colony Algorithm (ABC) |
| ۲۰۰۶ | Haddad et al. | honey-bee mating optimization |
| ۲۰۰۷ | Hamed Shah-Hosseini | Intelligent Water Drops |
| ۲۰۰۸ | Yang | firefly algorithm |
| ۲۰۰۸ | Mucherino and Seref | Monkey Search |
| ۲۰۰۹ | Yang and Deb | cuckoo search |
| ۲۰۰۹ | Yang | bat algorithm |
| ۲۰۱۱ | Metaheuristic Solutions | X algorithm. |

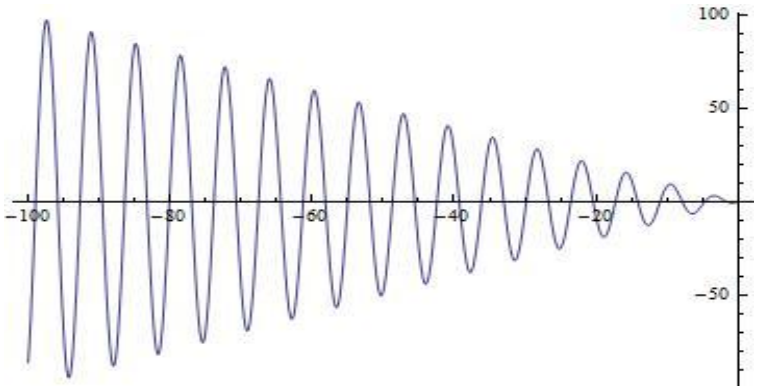
منبع داده‌ها: [۱۰]

1. Ant Colony Optimization(ACO)
2. Evolutionary Algorithms(EAs)
3. Simulated Annealing (SA)

## تولید جواب در فرا ابتکاری‌ها

نحوه تعیین همسایه‌ها از جواب فعلی، روش ارزیابی همسایه‌ها و چگونگی انتخاب همسایه جایگزین جواب فعلی در هر روش فرا ابتکاری، به گونه‌ای خاص انجام می‌شود. در بین روال‌های بهینه‌یابی[1]، روش‌های جواب‌یابی مکرر، نقش مهمی دارند. مهمترین گام در یک روال تکراری عبارتست از تولید جواب جدید $s'$ از جواب فعلی $s$ و آزمون اینکه این گام باید تکرار شود یا نه. تکنیک‌های جستجوی همسایگی[2] روال‌های تکراری هستند که در آنها همسایه $N(s)$ برای هر جواب $s$ تعریف می‌شود و جواب بعدی $s'$ از بین جواب‌های $N(s)$ انتخاب می‌شود [۵].

خین یائو[3] [۱۴] بر اساس تحقیقات خود بیان می‌دارد شبیه‌سازی تبریدی که از تعداد جواب همسایه بیشتری استفاده می‌کند، بهتر از موردی است که از تعداد همسایه کمتر بهره می‌گیرد. مشخص گردیده است که عملکرد واقعی شبیه‌سازی تبرید، بستگی بسیار زیادی به انتخاب پارامترها دارد. با این حال، تحقیقات زیادی در مورد پارامترهای مربوط به همسایگی، خصوصاً در کل فرایند شبیه‌سازی تبرید، انجام نشده است. لیو[4] از طریق آزمایش‌های محاسباتی، اثر اندازه همسایگی بر فرایند شبیه‌سازی تبرید در مسأله زمانبندی فلوشاپ را بررسی کرد و یک روال بهبودیافته شبیه‌سازی تبرید با اندازه متغیر همسایه را پیشنهاد کرد که در همه موارد بهتر، از روش سنتی شبیه‌سازی تبرید عمل می‌کرد [۸]. در ادامه فعالیت‌های تحقیقاتی برای توسعه شبیه‌سازی تبرید، لو، یوآن و ژانگ[5] [۹] روشی مبتنی بر تئوری ابر[6] را ارائه نمودند که امکان جستجوی بهتر همسایگی‌ها و دستیابی به جواب‌های بهتر را فراهم نمود.




نمودار۲. منحنی تابع $xCos(x)$ در بازه [۰ ۱۰۰-]

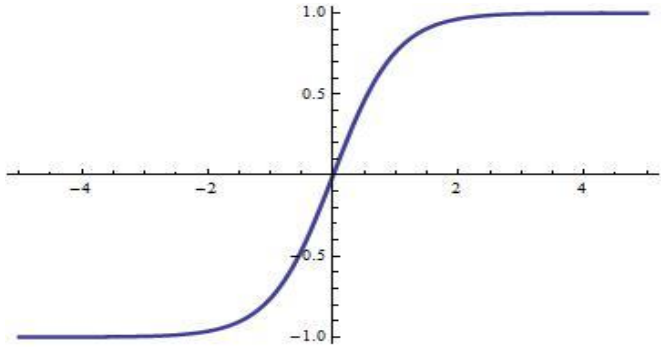
1. Optimization Procedures
2. Neighborhood Search Techniques
3. Xin Yao
4. Liu
5. Lu, Yuan and Zhang
6. Cloud Theory

### توابع مورد استفاده در تولید جواب

در روش پیشنهادی این مقاله، دو تابع xCos(x) و tanh(x)، نقش مهمی در تولید جوابهای همسایه دارند. در ادامه، ویژگیهای مهم این دو تابع بررسی می‌شود.

### الف) xCos(x)

این تابع، دارای محور تقارن x=۰ است. نمودار ۲، این تابع را در بازه [۰ ۱۰۰-] نشان می‌دهد. با نزدیک شدن مقدار x به صفر، دامنه نوسان مقدار تابع کاهش می‌یابد که از این خاصیت برای همگرا نمودن الگوریتم استفاده شده است.




### ب) tanh(x)

این تابع، دارای مرکز تقارن (۰ ۰) و برد (۱ ۱-) است. اگر عدد ثابتی مانند a در این تابع ضرب شود، برد آن شامل بازه باز (a a-) می‌شود. از این خاصیت برای تولید جوابهایی در فاصله اندک از جواب فعلی استفاده شده است.

### معرفی الگوریتم پیشنهادی

این الگوریتم، مبتنی بر جمعیتی از جواب‌هاست و مهمترین قسمت آن، فرمولهای تولید جواب‌های جدید از بهترین جواب فعلی است. دیگر ویژگی این الگوریتم، مشخص بودن تعداد تکرار به عنوان یک پارامتر ورودی است که باعث ثبات در عملکرد زمانی الگوریتم، یعنی انحراف معیار اندک در زمان‌های خاتمه الگوریتم در اجراهای متعدد می‌شود. جدول ۲ شبه‌کد الگوریتم را نشان می‌دهد.

*جدول۲. شبه‌کد الگوریتم پیشنهادی*

**مقداردهی پارامترها**

تعداد تکرار (K)
تعداد جواب‌های به‌دست آمده از فرمول اول ($n_1$)
تعداد جواب‌های به‌دست آمده از فرمول دوم ($n_2$)

**شروع**

$i \leftarrow 1$
$F^* \leftarrow +\infty$ (بهترین مقدار تابع هدف تا کنون)
کران بالا(U) و کران پایین(L) برای جواب‌ها را دریافت کن.
تعداد $n_1+n_2$ جواب اولیه را به صورت تصادفی ایجاد کن.
**حلقه** از $i=1$ تا $i=0/001$ با گام $\frac{0.999}{K}-$ انجام بده.

جواب‌های غیر موجه را تعمیر کن.
جواب‌ها را ارزیابی و مقدار تابع هدف به ازای هر جواب را پیدا کن.
بهترین مقدار تابع هدف از بین مقادیر فوق$\leftarrow F$
جواب مربوط به بهترین مقدار تابع هدف$\leftarrow S$
اگر $F \geq F^*$ ، آنگاه $F^* \leftarrow F$ و $S^* \leftarrow S$

تعداد $n_1$ جواب از طریق فرمول اول ( $S^* \times R \times iCos(100i)$ ) ایجاد کن. (R عدد تصادفی است.)
تعداد $n_2$ جواب از طریق فرمول دوم ( $S^* + (U - L) \times R \times 0.05tanh(iCos(100i))$ ) ایجاد کن.

**پایان حلقه**

$F^*$ و $S^*$ را چاپ کن.

**پایان**

## آزمون عملکرد الگوریتم

برای بررسی عملکرد الگوریتم پیشنهادی، از هفت تابع معروف در بهینه‌یابی پیوسته استفاده شد. این توابع را لو، یوآن و ژانگ [۹] به منظور بررسی کارایی الگوریتم شبیه‌سازی تبرید مبتنی بر تئوری ابر[1] به‌کار گرفتند. این توابع و دامنه بررسی آنها در جدول ۳ آمده‌اند. نمودار سه بعدی این توابع نیز در نمودارهای ۴ تا ۱۰ مشاهده می‌شود.

*جدول۳. مشخصات توابع مورد استفاده برای آزمون عملکرد الگوریتم*

| تابع | ضابطه تابع | دامنه تابع |
|---|---|---|
| Sphere model function | $F_1 = \sum_{i=1}^{5} x_i^2$ | $\|x_i\| \le 30$ |
| Hyper-ellipsoid function | $F_2 = \sum_{i=1}^{10} i^2 x_i^2$ | $\|x_i\| \le 1$ |
| Generalized Rastrigin's function | $F_3 = \sum_{i=1}^{30} \left( x_i^2 - 10\cos(2\pi x_i) + 10 \right)$ | $\|x_i\| \le 2.56$ |
| Branin function | $F_4 = \left( x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$ | $\|x_i\| \le 5$ |
| Griewangk's function | $F_5 = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | $\|x_i\| \le 256$ |
| DeJong's f2 function | $F_6 = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$ | $\|x_i\| \le 2.56$ |
| Six hump camel back function | $F_7 = 4x_1^2 - 2.1x_1^4 + x_1^6/3 + x_1 x_2 - 4x_2^2 + 4x_2^4$ | $\|x_i\| \le 5$ |

1. Cloud theory-based simulated annealing algorithm

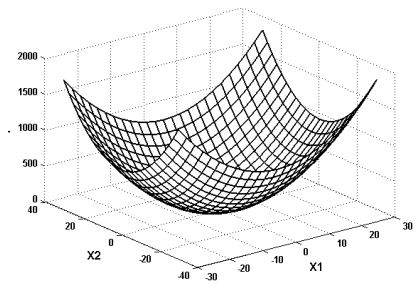
در بین توابع مورد بحث، F1 ، F2 و F3 پیوسته و تک قله‌ای[1] هستند، یعنی تنها یک بهترین جواب برای آنها وجود دارد. F4 پیوسته و چندقله‌ای است، یعنی دارای چند جواب بهینه است. چهار تابع مذکور برای آزمون قابلیت جستجوی الگوریتم به‌کار می‌روند.

F5 و F6 غیرخطی و دارای چند کمینه هستند. F5 دارای کمینه محلی در $i = 1, 2, ..., 256$ و $k = 1, 2, ..., 25($ ، $x_i = \pm k\pi\sqrt{i}$, است. F6 نقاط کمینه فراوانی در نزدیکی $x_1 = x_2 = 1$ دارد. مقادیر این دو تابع با تغییر x به صورت خطی تغییر نمی‌کنند و لذا از آنها برای آزمون توان فرار الگوریتم از افتادن در دام کمینه‌های محلی و رهایی از همگرایی زودرس استفاده شده است.

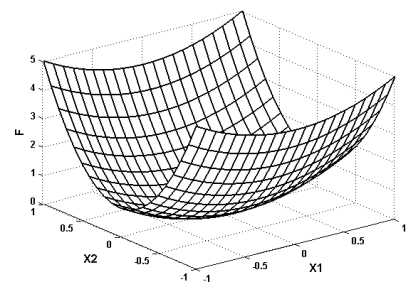
F7 یک تابع خاص غیرخطی چند کمینه است مقادیر آن، نه تنها کوچک بلکه نزدیک به یکدیگر هستند. به‌علاوه، این تابع، بینهایت کمینه محلی دارد که می‌تواند عملیات جستجو را به راحتی سد کند. دو نقطه (۰/۷۱۲۶۵۶ -۰/۰۸۹۸۴۲) و (۰/۷۱۲۶۵۶ ۰/۰۸۹۸۴۲-) نقاط کمینه سراسری این تابع هستند که F7 به ازای آنها دارای مقدار ۱/۰۳۱۶۲۸۵- خواهد شد. به خاطر دشوار بودن یافتن جواب بهینه برای این تابع، از آن برای آزمون توانایی الگوریتم در نزدیک شدن به بهینه سراسری و پایایی عملکرد آن استفاده شده است. شکل تابع در نمودار ۱۰ نشان داده شده است.

به منظور قابل مقایسه کردن الگوریتم پیشنهادی با الگوریتم‌های SA و CSA ، آزمایشهایی مشابه آنچه برای این دو الگوریتم انجام شده بود، طراحی و اجرا شد. ویژگیهای مشترک این آزمایش‌ها عبارت بودند از برنامه‌نویسی الگوریتم با نرم‌افزار MATLAB، 100 بار تکرار الگوریتم برای هر تابع با رایانه‌ای دارای پردازنده ۳ گیگاهرتز و RAM یک گیگابایت. مقدار پارامترهای الگوریتم نیز K=300 ، n1=10 و n2= 20 قرار داده شد. نتایج انجام آزمایش در مورد الگوریتم پیشنهادی به همراه نتایج گزارش شده توسط لو، یوآن و ژانگ در جدول ۴ آمده است.

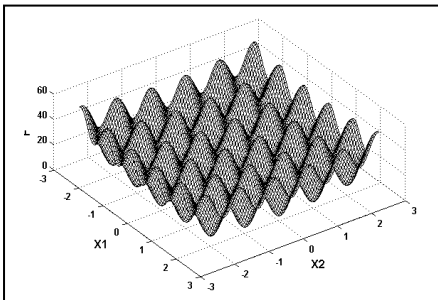
1. Single peak



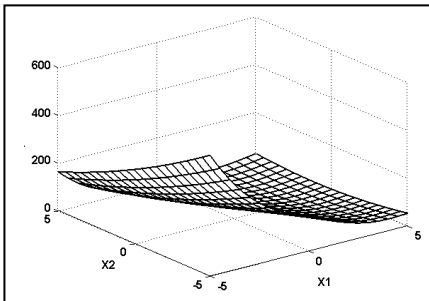

با فرض سه متغیره F1 نمودار ۴- شکل تابع



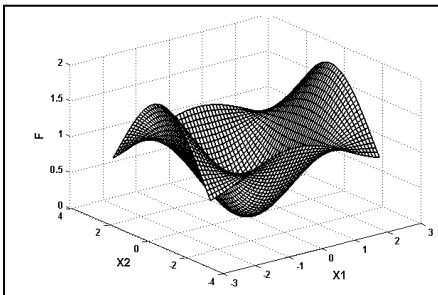

با فرض سه متغیره F2 نمودار ۵- شکل تابع



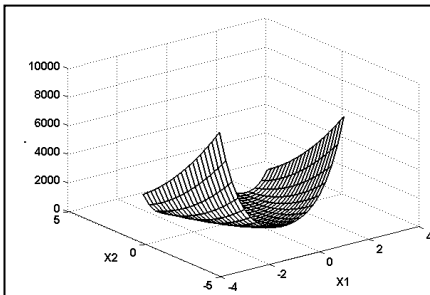

با فرض سه متغیره F3 نمودار ۶- شکل تابع



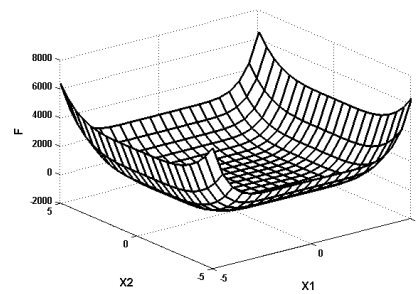

F4 نمودار ۷- شکل تابع




با فرض سه متغیره F5 نمودار ۸- شکل تابع




F6 نمودار ۹- شکل تابع




F7 نمودار ۱۰- شکل تابع

همان‌گونه که ملاحظه می‌شود، این الگوریتم از نظر سرعت و دقت، نسبت به دو الگوریتم دیگـر، برتری محسوسی دارد. به منظور بررسی آماری نتایج الگوریتم، فرضـیه اصـلی ایـن تحقیـق بـه صورت زیر بیان گردید:

«عملکرد الگوریتم پیشنهادی، بهتر از عملکرد الگوریتم‌های SA و CSA است.»

فرضیه‌های فرعی مربوط به کارایی الگوریتم پیشنهادی به شرح زیر است:

۱- جواب بهتری برای تابع F1 به دست می‌دهد.
۲- جواب بهتری برای تابع F2 به دست می‌دهد.
۳- جواب بهتری برای تابع F3 به دست می‌دهد.
۴- جواب بهتری برای تابع F4 به دست می‌دهد.
۵- جواب بهتری برای تابع F5 به دست می‌دهد.
۶- جواب بهتری برای تابع F6 به دست می‌دهد.
۷- جواب بهتری برای تابع F7 به دست می‌دهد.
۸- زمان کمتری جهت تولید جواب برای تابع F1 صرف می‌کند.
۹- زمان کمتری جهت تولید جواب برای تابع F2 صرف می‌کند.
۱۰-زمان کمتری جهت تولید جواب برای تابع F3 صرف می‌کند.
۱۱-زمان کمتری جهت تولید جواب برای تابع F4 صرف می‌کند.
۱۲-زمان کمتری جهت تولید جواب برای تابع F5 صرف می‌کند.
۱۳-زمان کمتری جهت تولید جواب برای تابع F6 صرف می‌کند.
۱۴-زمان کمتری جهت تولید جواب برای تابع F7 صرف می‌کند.

برای آزمون فرضیه‌های فرعی، از آزمـون t بـرای مقایسـه میـانگین دو جامعـه دارای واریـانس نامعلوم و نابرابر استفاده گردید. جامعه اول مورد بررسی را نتـایج الگـوریتم CSA قـرار دادیـم و جامعه دوم را نتایج الگوریتم پیشنهادی در نظر گرفتیم. با انجام محاسبات مربوط به آزمون آماری توسط نرم‌افزار Excel 2007، نتایج در جدول ۵ خلاصه گردید.

جدول۵. نتایج آزمون آماری فرضیه‌های فرعی

| فرضیه‌های فرعی | P-Value | نتیجه | فرضیه‌های فرعی | P-Value | نتیجه |
|---|---|---|---|---|---|
| ۱ | 3.44E-69 | مثبت | ۸ | 1.92E-117 | مثبت |
| ۲ | 6.29E-31 | مثبت | ۹ | 2.33E-119 | مثبت |
| ۳ | 1.07E-60 | مثبت | ۱۰ | 1.00E+00 | منفی |
| ۴ | 1.47E-09 | مثبت | ۱۱ | 8.00E-195 | مثبت |
| ۵ | 5.02E-20 | مثبت | ۱۲ | 8.92E-156 | مثبت |
| ۶ | 3.86E-14 | مثبت | ۱۳ | 3.97E-137 | مثبت |
| ۷ | 7.17E-28 | مثبت | ۱۴ | 4.11E-148 | مثبت |

همان‌گونه که ملاحظه می‌شود، به جز فرضیه شماره ۱۰، بقیه فرضیه‌ها در سطح اطمینان بالای ۹۹ درصد تأیید شدند. فرضیه شماره ۱۰ ـ که مربوط به کارایی زمانی الگوریتم در تابع شماره ۳ است ـ نیز با تغییر پارامترها و تنظیم دقیق آنها قابل تأیید است.

از طرف دیگر، « وانگ، لین و هوانگ» در ۲۰۱۰ در مقاله‌ای با عنوان « حل مسأله پشت شتر شش کوهانه با استفاده از الگوریتم تکاملی ترمودینامیکی» به حل تابع F7 پرداختند. آنها با تنظیم الگوریتم خود ـ به طوری که در هر تکرار ۱۰۰ جواب تولید شود و با ۱۵ بار تکرار آن ـ به نتایج مندرج در جدول ۶ دست یافتند.

جدول ۶. نتایج اجرای الگوریتم تکاملی ترمودینامیکی روی تابع F7

| F-min-val | X1 | X2 | Step |
|---|---|---|---|
| -1.031627816006554 | -0.089933 | 0.7123900 | 1997 |
| -1.031628310766583 | 0.0897910 | -0.712526 | 2197 |
| -1.031628417907029 | -0.089903 | 0.7127110 | 2155 |
| -1.031628453473775 | 0.0898420 | -0.712655 | 2289 |
| -1.031628452765956 | -0.089842 | 0.7126470 | 1847 |
| -1.031621705742450 | 0.0910380 | -0.712344 | 2539 |
| -1.031628453371039 | -0.089846 | 0.7126540 | 5000 |
| -1.031628445575145 | 0.0898690 | -0.712683 | 2399 |
| -1.031628115974580 | -0.090134 | 0.7127050 | 1885 |
| -1.031628443823104 | 0.0898920 | -0.712659 | 2152 |

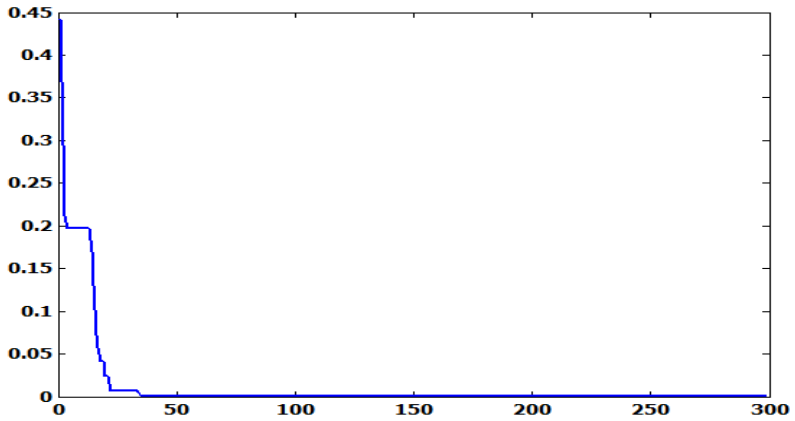
منبع: [۱۲]

با تحلیل نتایج و استخراج بدترین، بهترین و میانگین مقدار تابع از جدول مذکور و مقایسه آماری آن با نتایج الگوریتم پیشنهادی این مقاله مشخص می‌گردد که الگوریتم پیشنهادی، از حیث ارائه جواب نزدیک به بهینه، تفاوت معنی‌داری با الگوریتم « وانگ، لین و هوانگ» ندارد (P-value=0.35). جدول ۷ نتایج این مقایسه را نشان می‌دهد. شایان ذکر است که وانگ، لین و هوانگ، زمان دست‌یابی به نتایج توسط الگوریتم خود را ذکر ننموده‌اند، ولی با توجه به شواهد (تعداد گام‌های بسیار زیاد الگوریتم) زمان الگوریتم آنها بیشتر از زمان الگوریتم پیشنهادی به نظر می‌رسد.

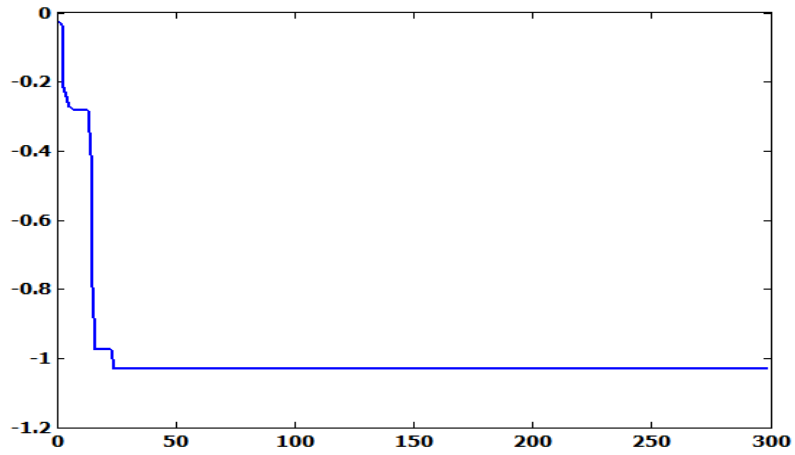
*جدول ۷. مقایسه نتایج اجرای الگوریتم تکاملی ترمودینامیکی و الگوریتم پیشنهادی در مورد تابع F7*

| استاندارد همه مقادیر | مقدار میانگین | بدترین مقدار | بهترین مقدار | |
|---|---|---|---|---|
| 2.10297E-06 | -1.031627662 | -1.031621706 | -1.031628453 | الگوریتم وانگ، لین و هوانگ |
| 7.5611E-07 | -1.031627917 | -1.031623874 | -1.031628453 | الگوریتم پیشنهادی |

یکی از موارد بیان‌کننده کارایی الگوریتم‌های فرا ابتکاری در گام‌های مختلف، نمودارهای همگرایی هستند که متخصصان فرا ابتکاری‌ها با استفاده از آنها می‌توانند قضاوت‌هایی درباره عملکرد الگوریتم در همگرایی زودرس یا تعداد مناسب تکرارهای الگوریتم داشته باشند. نمودارهای ۱۱ و ۱۲، دو نمونه از نمودارهای همگرایی الگوریتم پیشنهادی است. از آنجا که جواب‌های آغازین الگوریتم، به صورت تصادفی، از فضای جواب انتخاب می‌شوند، بهترین این جواب‌ها اغلب مقدار تابع هدف را به شکل نامطلوبی زیاد نشان می‌دهند (با فرض اینکه تابع هدف به صورت کمینه‌سازی باشد.). این امر باعث می‌شود نمودار همگرایی الگوریتم از مقدار بالا شروع شود و مقیاس نمودار، از نظر عمودی، بسیار افزایش یابد، به طوری که بهبودها در گام‌های بعدی چنان کوچک به نظر آیند که در نمودار قابل تشخیص نباشند.




نمودار ۱۱. نمودار همگرایی الگوریتم برای تابع F6




نمودار ۱۲. نمودار همگرایی الگوریتم برای تابع F7

**نتیجه‌گیری و پیشنهادها**

الگوریتم پیشنهادی، دارای این مزیت‌هاست: سرعت در اجرا، سادگی پیاده‌سازی و نزدیک بودن جواب‌های نهایی به بهینه سراسری. نتایج اجرای الگوریتم برای هفت تابع پیوسته معروف نشان داد که این الگوریتم در مقایسه با دو الگوریتم شبیه‌سازی تبرید و شبیه‌سازی تبرید مبتنی بر تئوری ابر، برتری کامل برخوردار دارد. از جمله مزایای روش فرا ابتکاری پیشنهادی می‌توان به موارد زیر اشاره کرد:

- ساده بودن ساختار الگوریتم
- سرعت بالای اجرای الگوریتم به دلیل تک حلقه بودن الگوریتم اجرا
- قابلیت فرار از بهینه‌های محلی
- قابلیت جستجوی گسترده فضای جواب
- پایایی الگوریتم (واریانس کم جوابهای نهایی اجراهای متعدد الگوریتم)
- مستتر بودن مکانیزم همگرایی الگوریتم در روش تولید جواب

الگوریتم فعلی را می‌توان در مورد سایر توابع پیوسته به‌کار برد و نتایج را با نتایج سایر الگوریتم‌ها مقایسه نمود. این الگوریتم همچنین قابلیت توسعه به منظور کاربرد در مسائل گسسته را دارد. به‌کارگیری این الگوریتم در مسائل بهینه‌یابی ترکیبیاتی واقعی، از جمله حوزه‌های بسیار وسیع قابل پیگیری توسط محققان است، زیرا گمان می‌رود مسائل قدیمی که با الگوریتم‌های قبلی حل گردیده‌اند، با استفاده از این الگوریتم، جواب‌های بهتری داشته باشند.

## منابع


1. Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv., 35*(3), 268-308. doi: 10.1145/937503.937505
2. Blum, C., & Roli, A. (2008). Hybrid Metaheuristics: An Introduction. In C. Blum, M. J. e. B. Aguilera, A. Roli & M. Sampels (Eds.), *Hybrid Metaheuristics* (pp. 1-30). Berlin: Springer-Verlag.
3. Borenstein, Y., & Poli, R. (2006). *Structure and metaheuristics*. Paper presented at the Proceedings of the 8th annual conference on Genetic and evolutionary computation, Seattle, Washington, USA.
4. Burke, E. K., & Kendall, G. (2005). Introduction. In E. K. Burke & G. Kendall (Eds.), Search Methodologies: *Introductory Tutorials in Optimization and Decision Support Techniques* (pp. 5-18). New York: Springer.
5. Ferland, J. A., Hertz, A., & Lavoie, A. (1996). An Object-Oriented Methodology for Solving Assignment-Type Problems with Neighborhood Search Techniques. *Operations Research, 44*(2), 347-359.
6. Gonzalez, T. F. (2007). Basic Methodologies. In T. F. Gonzalez (Ed.), *Handbook of Approximation Algorithms and Metaheuristics* (pp. 1.1-1.17). Boca Raton, FL, USA: Chapman and Hall/CRC.
7. Hendrix, E. M. T., & G.-Tóth, B. (2010). Goodness of optimization algorithms *Introduction to Nonlinear and Global Optimization* (Vol. 37, pp. 67-90): Springer New York.
8. Liu, J. (1999). The impact of neighbourhood size on the process of simulated annealing: Computational experiments on the flowshop scheduling problem. [doi: DOI: 10.1016/S0360-8352(99)00075-3]. *Computers & Industrial Engineering, 37*(1-2), 285-288.
9. LV, P., Yuan, L., & Zhang, J. (2009). Cloud theory-based simulated annealing algorithm and application. *Engineering Applications of Artificial Intelligence, 22*, 742–749.
10. Metaheuristics. (2011), 2011, from www.metaheuristic.com/metaheuristic_optimization.php
11. Talibi, E.-G. (2009). METAHEURISTICS: FROM DESIGN TO IMPLEMENTATION
12. Wang, X. (2010). *Solving Six-Hump Camel Back Function Optimization Problem by Using Thermodynamics Evolutionary Algorithm*.
13. Weise, T. (2007). Global Optimization Algorithm: Theory and Application
14. Xin, Y. (1991). Simulated Annealing with Extended Neighborhood. *International Journal of Computer Mathematics, 40*(3-4), 169-189.