

یک الگوریتم جدید کلونی مورچه مبتنی بر جمعیت برای بهینه‌سازی پیوسته

سحر کیان‌فر^۱، محمدرضا میبیدی^۲

^۱دانشگاه صنعتی امیرکبیر، آزمایشگاه محاسبات نرم، kianfar@aut.ac.ir

^۲دانشگاه صنعتی امیرکبیر، آزمایشگاه محاسبات نرم، mmeybodi@aut.ac.ir

چکیده- محدوده‌ی کاربرد بهینه‌سازی پیوسته بسیار وسیع است. بسیاری از مسائل و پردازش‌ها در دنیای واقعی به شکل مسائل بهینه‌سازی پیوسته مدل می‌شود. برای نمونه می‌توان مسئله طراحی اشکال بهینه در توربین‌ها، انتخاب مقدار پارامترهای پیوسته مثل دما و فشار در پروسه‌های صنعتی و آموزش شبکه عصبی برای تشخیص پزشکی را نام برد. الگوریتم‌های بهینه‌سازی کلونی مورچه دسته‌ای از الگوریتم‌های بهینه‌سازی تکاملی هستند که با موفقیت در حل مسائل بهینه‌سازی گسسته عمل کرده‌اند. اخیراً نسخه‌هایی از بهینه‌سازی کلونی مورچه برای مواجه با مسائل بهینه‌سازی پیوسته توسعه داده شده است. در این مقاله یک الگوریتم جدید کلونی مورچه پیوسته به نام $PbCACO$ برای حل مسائل بهینه‌سازی پیوسته ارائه می‌شود که بخوبی نقاط امید به جواب را در مسائل چند قله‌ای پیدا می‌کند. برای افزایش دقت، الگوریتم پیشنهادی با یک رویه‌ی جستجوی محلی ترکیب می‌شود. در انتها با انجام آزمایشاتی کارایی الگوریتم پیشنهادی مورد بررسی قرار می‌گیرد. کلید واژه- بهینه‌سازی پیوسته، الگوریتم کلونی مورچه، جستجوی محلی.

الگوریتم‌های کلونی مورچه مبتنی بر جمعیت را برای ساخت فرمون به کار می‌گیرند. از اینرو آن را $PbCACO$ می‌نامیم.

۱- مقدمه

در الگوریتم پیشنهادی یک کلونی از مورچه‌ها راه‌حل‌های جدید را می‌سازند. راه‌حل‌های به دست آمده از هر تکرار به تکرار بعدی می‌روند تا فرمون را بسازند. هر مورچه دارای حافظه‌ای است که بهترین راه‌حل بدست آمده‌اش را در آن ذخیره می‌کند، این حافظه برای ساخت فرمون بکار می‌رود. پروسه بهینه‌سازی با تعدادی راه‌حل تصادفی شروع می‌شود، هر مورچه در هر تکرار از الگوریتم با نمونه‌برداری از یک توزیع نرمال که ویژگی‌های آن فرمون اصلاح شده توسط مورچه‌ها هستند، یک راه‌حل جدید می‌سازد. راه‌حل‌های ساخته شده در هر تکرار بر اساس کیفیتشان حافظه‌ی مورچه‌ها را بروز می‌کنند. اگر راه‌حل جدید بهتر از راه‌حل موجود در حافظه مورچه‌ها باشد جایگزین آن می‌شود. این فاز همان بروز رسانی فرمون است که با هدف حرکت به سمت بهترین راه‌حل انجام می‌گیرد. الگوریتم پیشنهادی قدرت اکتشاف خوبی دارد و به راحتی نقاط امیدوار به جواب را پیدا می‌کند اما دقت جواب پایین است بنابراین نیاز به راهکاری برای تقویت راه حل به دست آمده از الگوریتم داریم. برای تقویت راه‌حل‌های خوب به دست آمده از الگوریتم یک رویه

هوش جمعی رویکرد نسبتاً جدیدی در حل مسائل بهینه‌سازی است که از رفتار اجتماعی حشرات و حیوانات دیگر الهام می‌گیرد، از جمله مورچه‌ها که برای تکنیک‌های زیادی مورد الهام قرار گرفته‌اند و بیشترین موفقیت را در تکنیک بهینه‌سازی به دست آورده‌اند، که به عنوان بهینه‌سازی کلونی مورچه (ACO) شناخته شده است [۱]. ACO از رفتار جستجوی غذای مورچه‌ها الهام گرفته شده است. این الگوریتم ابتدا برای حل مسائل بهینه‌سازی گسسته مانند مسئله فروشنده دوره‌گرد [۱-۳]، مسیریابی [۴-۶] و زمانبندی [۷، ۸] معرفی شد. در این مسایل متغیر با مجموعه محدود از مولفه‌ها مشخص می‌شود. ACO موفقیت زیادی را در زمینه بهینه‌سازی گسسته به دست آورد. باوجود این که ACO ابتدا برای مسائل بهینه‌سازی گسسته پیشنهاد شد، استفاده از آن برای حل مسائل بهینه‌سازی پیوسته مورد توجه قرار گرفت [۸، ۹]. در این دسته از مسائل بهینه‌سازی متغیر یک مقدار حقیقی از دامنه تعریف شده را می‌گیرد. الگوریتمی که در این مقاله ارائه می‌شود ایده‌ای مشابه

پروژه ساخت مشابه الگوریتم ACO_R [۱۵] انجام می‌گیرد اما نحوه‌ی ساخت ویژگی‌های توزیع نرمال متفاوت است. برای یک مسئله D بعدی، هر مورچه با ساخت D توزیع نرمال و نمونه‌برداری از آن‌ها یک راه حل با D بعد $(X_i = (x_1, x_2, \dots, x_D))$ را بصورت مولفه به مولفه می‌سازد. هر توزیع نرمال $N(\mu_i, \sigma_i)$ برای بعد i ام دارای ویژگی‌های میانگین μ_i و واریانس σ_i است. برای ساخت ویژگی‌های توزیع، هر مورچه نیاز به یک راه حل راهنما (X_{guide}) دارد.

برای انتخاب راه حل راهنما به هر مورچه i یک احتمال Pc_i تعلق می‌گیرد. هر مورچه با احتمال Pc_i بهترین راه حل سراسری را به عنوان راهنما انتخاب می‌کند و با احتمال $1 - Pc_i$ بهترین راه حل موجود در حافظه‌اش را به عنوان راهنما در نظر می‌گیرد (رابطه ۱).

$$X_{guide_i} = \begin{cases} X_{best} & \text{if } rand(0,1) > Pc_i \\ X_i & \text{else} \end{cases} \quad (1)$$

مقادیر مختلف Pc برای مورچه‌ها منجر به تنوع بیشتر جمعیت می‌شود و نتایج بهتری را برای توابع چندقله‌ای نشان می‌دهد. با انتخاب Pc متفاوت برای هر مورچه، هر مورچه سطح متفاوتی از اکتشاف و استخراج خواهد داشت، بنابراین تنوع جمعیت افزایش می‌یابد. بدین منظور برای محاسبه احتمال Pc مورچه i از رابطه (۲) استفاده می‌کنیم [۱۷].

$$Pc_i = 0.05 + 0.45 * \frac{\left(\exp\left(\frac{10(i-1)}{ps-1}\right) - 1 \right)}{(\exp(10) - 1)} \quad (2)$$

مولفه‌های راه حل راهنما به عنوان میانگین توزیع نرمال در نظر گرفته می‌شوند. داریم:

$$\mu_i = \{\mu_i^1, \mu_i^2, \dots, \mu_i^D\} = \{x_{guide_i}^1, x_{guide_i}^2, \dots, x_{guide_i}^D\} = X_{guide_i}$$

واریانس توزیع i ام، برابر است با فاصله راه حل راهنما از بهترین راه حل مورچه‌های دیگر در بعد i ام ضربدر پارامتر ξ (رابطه (۳) [۱۵]).

$$\sigma_i^j = \xi \sum_{l=1}^m \frac{|x_{guide_i}^j - x_l^j|}{m-1} \quad (3)$$

پارامتر $\xi > 0$ ، برای همه‌ی ابعاد یکسان است و تأثیری مشابه نرخ تبخیر فرمون در بهینه سازی کلونی مورچه دارد. مقادیر بالاتر ξ سرعت همگرایی پایین الگوریتم را به همراه دارد. نرخ تبخیر فرمون حافظه بلند مدت را تحت تأثیر قرار می‌دهد یعنی راه‌های بدتر سریعتر فراموش می‌شوند. ξ نیز حافظه بلند مدت رو تحت تأثیر قرار می‌دهد، یعنی ξ هر چه بزرگتر باشد، جستجو کمتر به سمت نقاطی از فضای جستجو که استخراج

جستجوی محلی به الگوریتم اضافه می‌کنیم. برای جستجوی محلی از الگوریتم HJ [۱۰] استفاده می‌کنیم. در هر تکرار از الگوریتم جستجوی محلی به بهترین راه حل کلونی اعمال می‌شود. پروژه جستجوی محلی بعد از رسیدن به ماکزیمم تعداد ارزیابی تابع هدف متوقف می‌شود. برای نشان دادن کارایی الگوریتم پیشنهادی آن را بر روی مجموعه توابع تست ارائه شده در نشست بهینه‌سازی پیوسته کنگره محاسبات تکاملی IEEE در سال ۲۰۰۵ (CEC2005) [۱۱] اجرا می‌کنیم.

ادامه مقاله به صورت ذیل سازماندهی شده است. در بخش دوم الگوریتم‌های کلونی مورچه پیوسته ارائه شده تا کنون را مرور می‌نمایم. در بخش سوم الگوریتم پیشنهادی را به همراه جزئیات پیاده سازی ارائه می‌دهیم و در بخش چهارم نتایج شبیه سازی ارائه می‌گردد. در نهایت در بخش پنجم نتیجه گیری ارائه می‌شود.

۲- الگوریتم کلونی مورچه برای مسائل بهینه‌سازی پیوسته

الگوریتم‌های بهینه‌سازی زیادی هستند که ابتدا برای بهینه‌سازی ترکیبی توسعه داده شده‌اند و بعدها برای بهینه سازی مسائل پیوسته تعمیم یافته‌اند. مثل الگوریتم ژنتیک پیوسته^۳، تابکاری فلزات افزایشی^۴ و جستجوی Tabu پیوسته افزایشی^۵. برای تعمیم الگوریتم کلونی مورچه برای بهینه سازی توابع پیوسته سه رویکرد کلی به کار رفته است. الگوریتم‌های ارائه شده در دسته اول به صورت مستقیم از رفتار مورچه‌ها الهام گرفته‌اند [۸، ۱۲]. ولی دو رویکرد دیگر بهینه‌سازی کلونی مورچه ارائه شده برای حل مسائل گسسته را به فضای پیوسته تعمیم داده‌اند، این تعمیم با دو روش انجام گرفته: گسسته‌سازی فضای جستجو [۱۳، ۱۴] و نمونه‌برداری احتمالاتی از فضای جستجو [۱۵، ۱۶].

۳- الگوریتم PbCACO

ابتدا m مورچه بطور یکنواخت در فضای جستجو توزیع می‌شوند. الگوریتم PbCACO یک الگوریتم تکراری است که در سه فاز ساخت راه حل، بروزرسانی فرمون و جستجوی محلی تکرار می‌شود. جزئیات هر فاز را در ادامه بیان می‌نمایم.

³ Continuous Genetic Algorithms (CGA)

⁴ Enhanced Simulated Annealing (ESA)

⁵ Enhanced Continuous Tabu search (ECTS)

شده اند، بایاس می‌شود.

برای هر مورچه، با استفاده از یک تولید کننده تصادفی نرمال مانند جعبه مولر^۶، با میانگین μ_i^j و انحراف معیار σ_i^j مقدار λ مین مولفه از λ ام متغیر مشخص می‌شود، این پروسه تا انتخاب m مین متغیر تصمیم تکرار می‌شود و در انتهای پروسه‌ی ساخت، راه‌حل ساخته شده به سمت بهترین راه‌حل کلونی حرکت می‌کند (رابطه (۴)).

$$X_{new}^i = X_{new}^i + rand(0,1)(X_{best}^i - X_{new}^i) \quad (4)$$

در نهایت مقدار تابع شایستگی برای راه‌حل‌های جدید محاسبه می‌شود. به تعداد مورچه‌ها راه حل جدید ساخته می‌شود. در صورتی که راه‌حل جدید ساخته شده توسط مورچه‌ی m بهتر از راه‌حل موجود در حافظه‌اش باشد، حافظه مورچه با استفاده از راه‌حل جدید بروز می‌شود.

الگوریتم PbCACO نقاط امید به جواب را پیدا می‌کند برای افزایش دقت الگوریتم از یک جستجوی محلی برای تقویت راه‌حل‌های پیدا شده توسط الگوریتم استفاده می‌کنیم. برای این منظور از الگوریتم جستجوی محلی HJ^[۱۰] استفاده می‌کنیم. در هر تکرار از الگوریتم پروسه‌ی جستجوی محلی برای بهترین مورچه کلونی فراخوانی می‌شود. پروسه جستجوی محلی بعد از رسیدن به ماکزیمم تعداد ارزیابی تابع هدف متوقف می‌شود. تمرکز روی جستجو حول بهترین راه‌حل پیدا شده تا کنون است. برای رعایت محدوده مجاز متغیرها رابطه (۵) دنبال می‌شود.

$$x_i^d = \min(x_{max}^d, \max(x_{min}^d, x_i^d)) \quad (5)$$

۳-۱- جستجوی محلی

HJ یک روش جستجوی الگو^۸ است که در سال ۱۹۶۱ توسط هوک و جیوز^۹ معرفی شد [۱۰]. علی رغم گذشت زمان طولانی از ارائه الگوریتم، هنوز هم در بسیاری از مسائل جستجوی محلی قطعی به عنوان انتخاب اول بکار می‌رود. HJ راه‌حل به دست آمده از جستجوی سراسری را به عنوان نقطه پایه انتخاب می‌کند و در دو فاز اکتشاف و حرکت الگو آن را تقویت می‌کند. الگوریتم با یک حرکت اکتشافی آغاز می‌شود که در آن همه‌ی متغیرهای نقطه پایه به نوبت با یک گام از پیش تعریف شده (s_0) حرکت می‌کند. از آنجایی که هر دو جهت محور مختصات

متغیرها شانس یکسانی برای بهبود دارد این حرکت در دو جهت انجام می‌گیرد، ابتدا در جهت از پیش تعریف شده حرکت می‌کند و در صورت عدم بهبود، حرکت در جهت مخالف انجام می‌گیرد. بنابراین، این فاز حداقل نیاز به D و حداکثر 2D ارزیابی تابع شایستگی دارد که D تعداد ابعاد است. سپس حرکت الگو همه‌ی تغییرات موفق فاز اکتشاف را همزمان تکرار می‌کند. نتیجه حرکت ارزیابی می‌شود و در صورت موفق بودن حرکت نقطه جدید به عنوان نقطه پایه به کار می‌رود. این پروسه تا زمانی که بهبودی در هیچ یک از ابعاد حاصل نشود، تکرار می‌شود. اندازه گام‌های بکار رفته در طول زمان کاهش می‌یابد. ما در آزمایشات کاهش نمایی گام را بکار گرفته‌ایم.

$$s_j = s_{j-1} * b^j \quad (6)$$

اندازه گام اولیه (s_0) با توجه به دانش اولیه از فضای جستجو مشخص می‌شود. مقدار b از مجموعه $\{0.2, 0.3, 0.5\}$ انتخاب می‌شود. شبه کد جستجوی محلی در شکل ۱ نشان داده شده است.

Hooke-Jeeves Pattern Search Algorithm	
1.	Obtain initial base point x^t . Determine set of step lengths
2.	Move the base point along every one of the d dimensional axes at a time and evaluate the result. Adopt each new point if improvement on the previous point. This takes at least d , at most $2d$ evaluations. If any of the moves was successful, go to 3. If none was successful, go to 4.
3.	Repeat the successful moves in a combined pattern move. If the new point has a better fitness, assume it as the new base point. Return to 2 whichever the outcome.
4.	Adjust step length to next smaller step. If there is a smaller step, continue from 2. If not, terminate.

شکل ۱. شبه کد الگوریتم جستجوی محلی HJ [۱۰]

۴- آزمایشات

تاکنون الگوریتم‌های بهینه‌سازی بسیار زیادی طراحی شده و برای حل مسائل بهینه‌سازی پیوسته به کار رفته‌اند. در بیشتر کارهای انجام شده، زیر مجموعه‌ای از مسائل استاندارد مانند اسفیر، رزنبراک، رسترچین و ... برای ارزیابی الگوریتم پیشنهادی به کار رفته است. هر کدام از این الگوریتم‌ها روی برخی از توابع به خوبی عمل می‌کنند ولی روی توابع دیگر چنین کارایی را ندارند. از این رو نیاز به تعریف استاندارد از معیارهای توقف مسئله، اندازه مسئله، مقداردهی اولیه و سایر رفتارهای الگوریتم داریم، تا بتوانیم الگوریتم‌ها را در شرایط یکسانی ارزیابی کنیم. در سال ۲۰۰۵ مجموعه‌ای از توابع محک استاندارد به نام CEC2005 برای فراهم آوردن معیارهای ارزیابی یکسان ارائه

⁶ Box-muller

⁷ Hooke-Jeeves

⁸ Pattern search

⁹ Hooke and Jeeves

۵- نتیجه‌گیری

در این مقاله یک الگوریتم جدید کلونی مورچه به نام PbCACO ارائه نمودیم که با انتقال جمعیت مورچه‌ها به تکرار بعدی فرمون را می‌سازد. قدرت اکتشاف الگوریتم ارائه شده بالا است، برای افزایش دقت الگوریتم یک روند جستجوی محلی به آن افزودیم. الگوریتم را بر روی توابع تست ارائه شده در CEC2005 اجرا کردیم. نتایج بدست آمده کارایی خوب الگوریتم پیشنهادی در محیط‌های پیوسته با ویژگی‌های متفاوت را نشان داد. نتایج نشان داد که الگوریتم به خوبی در محیط‌های چند قله‌ای و چرخش یافته بهینه را می‌یابد.

۶- تقدیر و تشکر

قسمتی از این کار تحقیقاتی با حمایت مالی مرکز تحقیقات مخابرات ایران انجام گرفته است.

مراجع

- [1] M. Dorigo, V. Maniezzo, and A. Colomi, "Ant system: Optimization by a colony of cooperating agents," *IEEE Transactions on Systems*, pp. 29-41, 1996.
- [2] L. M. G. Marco Dorigo "Ant Colonies for the Traveling Salesman Problem," *BioSystems*, vol. 43, pp. 73-81, 1997.
- [3] L. M. G. Marco Dorigo "Ant Colony System: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation* vol. 1, pp. 53-66, 1997.
- [4] B. Bullnheimer, R. F. Hartl, and C. Strauss, "Applying the Ant System to the Vehicle Routing Problem," in *2nd International Conference on Metaheuristics*, pp. 1-12, 1997.
- [5] R. F. H. Bernd Bullnheimer, Christine Strauss "An Improved Ant System Algorithm for the Vehicle Routing Problem" *Annals of Operations Research* vol. 89, pp. 319-328, 1999.
- [6] I. L. Silvia Mazzeo, "An Ant Colony Algorithm for the Capacitated Vehicle Routing," in *Electronic Notes in Discrete Mathematics*, pp. 181-186, 2004.
- [7] J. Zhang, X. Hu, X. Tan, J. H. Zhong, and Q. Huang, "Implementation of an Ant Colony Optimization Technique for Job Shop Scheduling Problem," *Transactions of the Institute of Measurement and Control* vol. 28, pp. 93-108, 2006.
- [8] J. Dréo and P. Siarry, "Continuous interacting ant colony algorithm based on dense heterarchy," *Future Generation Computer Systems*, vol. 20, pp. 841-856, 2004.
- [9] L. Jiao, L. Wang, X. Gao, J. Liu, F. Wu, M. Kong, and P. Tian, "Application of ACO in Continuous Domain," in *Advances in Natural Computation*. vol. 4222: Springer Berlin / Heidelberg, pp. 126-135, 2006.
- [10] R. Hooke and T. A. Jeeves, "Direct Search Solution of Numerical and Statistical Problems," *J. ACM*, vol. 8, pp. 212-229, 1961.
- [11] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 Special Session on Real Parameter Optimization," Nanyang Technological University 2005.
- [12] G. Bilchev and I. C. Parmee, "The ant colony metaphor for searching continuous design spaces," in *Proceedings of the AISB Workshop on Evolutionary Computation*. vol. 993 London, UK Springer-Verlag, pp. 25-39, 1995.
- [13] L. Yan-jun and W. Tie-jun, "An adaptive ant colony system algorithm for continuous-space optimization problems," *Journal of Zhejiang University - Science A*, vol. 4, pp. 40-46, 2003.

شد [۱۱]. ما از این مجموعه برای ارزیابی الگوریتم‌های ارائه شده در مقاله استفاده می‌نمایم. برای اطلاع از ویژگی‌ها و فرمول ریاضی ۲۵ تابع تست موجود در این مجموعه به مرجع [۱۱] مراجعه نمایید توابع f1-f5 تک قله‌ای هستند و توابع f6-f25 چندقله‌ای اند. توابع f1-f11، f14 و f16-f25 چرخش^{۱۱} یافته‌اند. ما از توابع f1-f16 برای آزمایشات استفاده می‌کنیم. معیار توقف الگوریتم $1e5 \times D$ ارزیابی تابع است که D تعداد ابعاد است و آزمایشات برای ابعاد ۱۰ و ۳۰ انجام می‌گیرد. الگوریتم در محیط MATLAB 7.6.0 پیاده‌سازی شده‌است. آزمایشات ۳۰ بار تکرار شده‌اند و متوسط نتایج و انحراف معیار آن‌ها گزارش شده است.

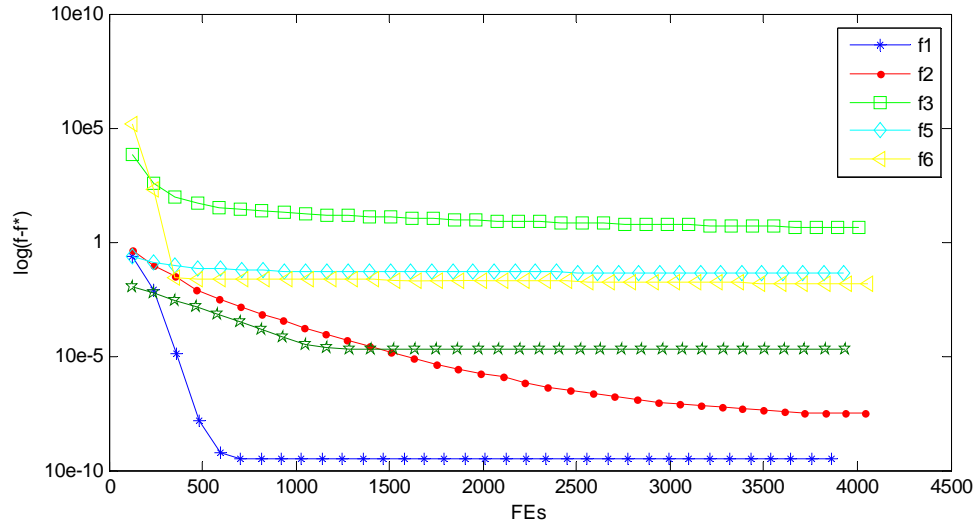
تعداد مورچه‌ها برای ۱۰ بعد برابر ۱۰ و برای ۳۰ بعد برابر ۴۰ در نظر گرفته شده است. نرخ تبخیر فرمون τ برابر ۰.۸۵ در نظر می‌گیریم. پارامتر ماکزیمم گام جستجوی محلی را برابر ۵ و اندازه گام اولیه یعنی s_0 را برابر ۴ در نظر گرفتیم. ماکزیمم ارزیابی تابع برای جستجوی محلی را برابر ۱۰۰ در نظر می‌گیریم. مقادیر خطای به دست آمده یعنی $(f - f^*)$ ، در جدول ۱ آورده شده است. در این جدول کمترین مقدار، بیشترین مقدار، میانگین و واریانس ۳۰ بار اجرای الگوریتم را گزارش نمودیم. نمودار همگرایی توابع تست در شکل ۲ و شکل ۳ نشان داده شده است.

همانطور که نتایج نشان می‌دهد الگوریتم نه تنها جواب خوبی برای توابع تک قله‌ای (f1-f5) می‌دهد، بلکه به خوبی نقاط بهینه را در توابع چندقله‌ای و چرخش یافته پیدا می‌کند. زیرا مکانیزم ارائه شده برای انتخاب راه‌حل رانما تنوع را برای مدت زیادی حفظ می‌کند و از همگرایی سریع الگوریتم به بهینه محلی جلوگیری می‌کند. برای توابع چرخش یافته نیز چون الگوریتم PbCACO وابستگی بین ابعاد را در نظر می‌گیرد، نتایج خوبی به دست می‌آید.

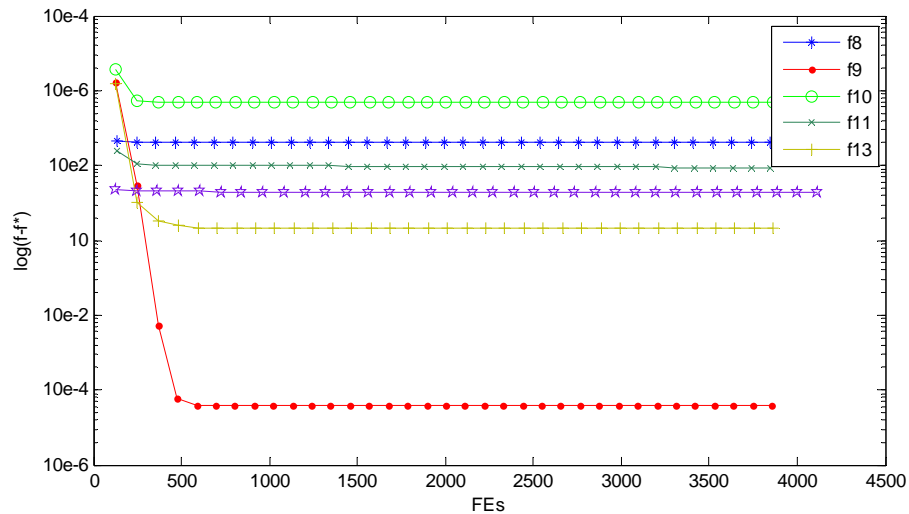
برای نشان دادن کارایی الگوریتم در مقایسه با الگوریتم‌های دیگر آن‌را با دیگر الگوریتم‌های موفق [18] CMA-ES و [19] adaptiveDE مقایسه کردیم. نتایج این مقایسه در جدول ۲ آورده شده است. نتایج نشان می‌دهد که الگوریتم PbCACO با تعداد جمعیت کمتر نتایج بهتری را نشان می‌دهد.

^{۱۱} کد متلب، جاوا و c این مجموعه در وب سایت <http://www.ntu.edu.sg/home/EPNSugan/> قابل دسترسی است
rotated

- [17] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *Evolutionary Computation, IEEE Transactions on*, vol. 10, pp. 281-295, 2006.
- [18] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, Vol. 2, pp. 1769-1776, 2005.
- [19] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Evolutionary Computation, 2005The 2005 IEEE Congress on*, Vol. 2, pp. 1785-1791, 2005.
- [14] M. Kong, P. Tian, Y. Hao, J. Liu, Y. Wang, Y.-m. Cheung, H. Yin, L. Jiao, J. Ma, and Y.-C. Jiao, "A Binary Ant Colony Optimization for the Unconstrained Function Optimization Problem Computational Intelligence and Security." vol. 3801: Springer Berlin / Heidelberg, pp. 682-687, 2005.
- [15] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *European Journal of Operational Research*, vol. 185, pp. 1155-1173, 2008.
- [16] S. Pourtakdoust, H. Nobahari, M. Dorigo, M. Birattari, C. Blum, L. Gambardella, F. Mondada, and T. Stützle, "An Extension of Ant Colony System to Continuous Optimization Problems," in *Ant Colony Optimization and Swarm Intelligence*. vol. 3172: Springer Berlin / Heidelberg, pp. 158-173, 2004.



شکل ۲. نمودار همگرایی توابع f1-f6



شکل ۳. نمودار همگرایی توابع f8-f13

جدول ۱. نتایج بدست آمده از اجرای PbCACO روی F1-F16

D=30				D=10				
واریانس	میانگین	بیشترین	کمترین	واریانس	میانگین	بیشترین	کمترین	
3.84E-01	2.51E-06	3.34E+00	1.52E-06	1.61E-05	3.13E-10	7.69E-05	0	F1
4.18E+02	9.25E-01	1.93E+03	3.59E-01	2.31E-03	3.33E-08	8.19E-03	5.06E-09	F2
1.86E+06	5.86E+06	9.69E+06	2.61E+06	5.87E+04	7.98E+01	2.43E+05	3.57E+01	F3
2.56E+00	1.08E+01	1.74E+01	4.73E+00	1.13E-05	4.19E-05	6.15E-05	2.18E-05	F4
2.85E+03	1.07E+04	1.82E+04	5.25E+03	2.48E+03	3.54E-05	1.04E+01	1.71E-5	F5
1.32E+03	1.79E+00	6.47E+03	5.06E-01	2.81E+02	1.66E+00	9.88E+02	2.25E+00	F6
1.10E-01	6.57E-01	8.84E-01	3.95E-01	1.15E+00	1.47E+00	4.90E+00	4.94E-02	F7
5.34E-02	2.10E+01	2.11E+01	2.08E+01	1.20E-01	2.03E+01	2.06E+01	2.01E+01	F8
2.10E+01	1.77E-01	2.10E+02	1.17E-01	3.33E-03	5.97E-03	1.46E-02	8.40E-04	F9
3.88E+01	3.10E+00	3.87E+02	2.24E+00	1.86E+01	5.41E-02	1.19E+02	2.02E-02	F10
1.12E+00	4.10E+00	4.23E+01	3.71E+00	1.10E+00	8.55E-01	1.09E+01	6.21E-01	F11
5.31E+04	2.00E+04	3.32E+05	1.28E+03	1.19E+03	5.93E+02	6.06E+03	2.91E+00	F12
7.22E+00	2.84E+00	5.50E+01	2.03E+00	7.34E-01	1.45E-01	3.60E+00	4.52E-01	F13
3.56E-01	1.33E+01	1.40E+01	1.26E+01	4.06E-01	3.98E+00	4.50E+00	2.84E+00	F14
1.07E+02	6.36E+02	9.11E+02	4.86E+02	2.18E+02	3.11E+02	8.00E+02	1.41E-02	F15
8.31E+01	4.12E+01	5.55E+02	2.98E+01	5.08E+01	2.31E+01	3.63E+02	1.45E+02	F16

جدول ۲. مقایسه الگوریتم PbCACO با الگوریتم‌های دیگر

d=30		d=10			
restart CMA-ES	PbCACO	adaptiveDE	restart CMA-ES	PbCACO	
5.42e-9	2.51E-06	0.00E+00	5.20e-09	3.13E-10	F1
6.22e-9	9.25E-01	5.70E-02	4.70e-09	3.33E-08	F2
5.55e-9	5.86E+06	2.09E+05	5.60e-09	7.98E+01	F3
1.11E+04	1.08E+01	6.19E-01	5.02e-09	4.19E-05	F4
8.62e-9	1.07E+04	1.22E+02	6.58e-09	3.54E-05	F5
5.90e-9	1.79E+00	3.54E+00	4.87e-09	1.66E+00	F6
5.31e-9	6.57E-01	1.58E+02	3.31e-09	1.47E+00	F7
2.01E+01	2.10E+01	2.04E+01	2.00E+01	2.03E+01	F8
9.38e-1	1.77E-01	3.97E+00	2.39e-01	5.97E-03	F9
1.65E+00	3.10E+00	1.37E+01	7.96e-02	5.41E-02	F10
5.48E+00	4.10E+00	2.20E+00	9.34e-01	8.55E-01	F11
4.43E+04	2.00E+04	2.62E+02	2.93E+01	5.93E+02	F12
2.49E+00	2.84E+00	5.98E-01	6.96e-01	1.45E-01	F13
1.29E+01	1.33E+01	3.21E+00	3.01E+00	3.98E+00	F14
2.08E+02	6.36E+02	1.84E+02	2.28E+02	3.11E+02	F15
3.50E+01	4.12E+01	1.22E+02	9.13E+01	2.31E+01	F16