مجله محاسبات نرم، جلد 11، شماره 2 (پاییز و زمستان 1401)، صفحه 62ـ71
https://doi.org/10.22052/scj.2023.248427.1104

مجله محاسبات نرم
SOFT COMPUTING JOURNAL
تارنمای مجله: scj.kashanu.ac.ir

University of Kashan

الگوریتم فراابتکاری نفرون-2 (NOA-2)، جهت حل مسائل بهینه‌سازی ✧

رضا بهمنش[1]، استادیار، نگار مجمع[2*]، استادیار

[1] گروه مهندسی صنایع، مؤسسه آموزش عالی نقش‌جهان، اصفهان، ایران.

[2] گروه مهندسی کامپیوتر، مؤسسه آموزش عالی نقش‌جهان، اصفهان، ایران.

اطلاعات مقاله

تاریخچه مقاله:
دریافت 19 مهر ماه 1401
پذیرش 17 فروردین ماه 1402

کلمات کلیدی:
بهینه‌سازی
الگوریتم نفرون
فراابتکاری
تنوع‌بخشی
تمرکزگرایی

چکیده

امروزه الگوریتم‌های بهینه‌سازی فراابتکاری در حل مساله‌های بهینه‌سازی محبوبیت فراوانی پیدا کرده‌اند. با استفاده از این دسته الگوریتم‌های می‌توان به راحتی و به دور از پیچیدگی بسیاری از مساله‌های حوزه مهندسی را حل نمود. الگوریتم بهینه‌سازی نفرون-2 (NOA-2) نیز از این دسته الگوریتم‌ها است که توسعه اولین نسخه الگوریتم نفرون است. این الگوریتم الهامی از عملکرد نفرون در کلیه انسان است. ساختار الگوریتم NOA-2 که در این مقاله پیشنهاد شده، طبق رفتار نفرون مشتمل بر 4 بخش: جداسازی، جذب، تراوش و دفع است. برای ارزیابی عملکرد، به بررسی نتیجه اجرای الگوریتم NOA-2 و پنج الگوریتم بهینه‌سازی معروف دیگر بر روی هفت مساله بهینه‌سازی پرداخته شده است. در این ارزیابی، دو معیار کیفیت جواب (تابع هدف) و زمان حل محاسباتی برای ارزیابی و مقایسه در نظر گرفته شده‌اند. نتایج نشان می‌دهد که الگوریتم NOA-2 نسبت به سایر الگوریتم‌ها بهترین تابع هدف را در زمان معینی یافته و همچنین در زمان کمتری نسبت به سایر الگوریتم‌ها جواب بهینه هفت مساله مورد مطالعه را به دست آورده است.

1. مقدمه

امروزه، با افزایش پیچیدگی در مساله‌های بهینه‌سازی در حوزه ریاضیات و علوم کامپیوتر، الگوریتم‌های بهینه‌سازی فراابتکاری جدید مطرح شده‌اند. این روش‌ها می‌توانند جواب‌های به نسبت خوبی را برای مساله‌های مذکور تامین کنند. مطالعات و تحقیقات گسترده‌ای به معرفی الگوریتم‌های الهام گرفته شده از طبیعت پرداخته‌اند [1]-[2]. با وجود بکارگیری روش‌های فراابتکاری در حل مساله‌های بهینه‌سازی ترکیباتی نظیر جدول زمانی[1] [3]، زمان‌بندی [4]، برنامه‌ریزی تولید [5]، هنوز هم راه‌هایی برای شبیه‌سازی الگوریتم‌ها طبق رفتار طبیعی حیوانات یا اعضا بیولوژیکی وجود دارد که بتوان از طریق آنها ساختار کنونی الگوریتم‌ها را بهبود داد. یکی از این الگوریتم‌ها، الگوریتم نفرون است. در این مقاله بهبودی بر نسخه اولیه این الگوریتم که توسط نویسندگان در کارهای پیشین [6]، انجام شده است، ارائه می‌شود.

الگوریتم نفرون، الهامی از ساختار نفرون در کلیه انسان است. در واقع، در کلیه انسان، واحدهای متعددی به نام نفرون وجود دارد

✧ نوع مقاله: پژوهشی

* نویسنده مسئول

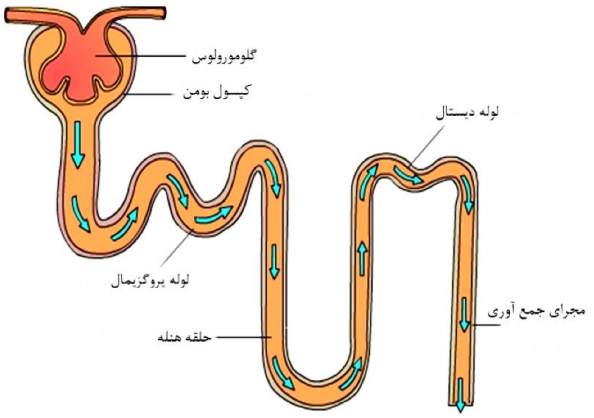
پست(های) الکترونیک: r.behmanesh@naghshejahan.ac.ir (بهمنش)

Majma@naghshejahan.ac.ir (مجمع)

[1] Timetabling

نحوه ارجاع به مقاله: بهمنش، رضا، مجمع، نگار، «الگوریتم فراابتکاری نفرون-2 (NOA-2)، جهت حل مسائل بهینه‌سازی»، مجله محاسبات نرم، جلد 11، شماره 2، ص62ـ71، پاییز و زمستان 1401.

که هر کدام شامل یک عضو فیلترکننده[1]، یک لوله[2] برای جذب[3] و تراوش[4] است که در شکل (1) مشاهده می‌شود، در نهایت نتیجه عملکرد این اعضا با یکدیگر، تولید ادرار است.



شکل (1): ساختار یک نفرون

بخش اول، ابتدا ذرات بزرگ را از خون جدا کرده و در ادامه آب و ذرات کوچک را به بخش دوم حمل می‌کند. در این بخش، با جذب مواد مغذی ضروری و دفع مواد غیرضروری به عنوان ادرار، حجم و فشارخون تنظیم می‌گردد [7]. علاوه بر این، بافتی به‌نام گلومورولوس[5] در ابتدای نفرون قرار دارد و 20% پلاسمای خون را درون نفرون وارد می‌کند و تمامی مواد در کپسول بومن[6] فیلتر می‌شوند. قانون دارسی[7] یک رابطه عمومی است که نرخ تخلیه[8] را به صورت تابعی از سطح جریان قابل‌عبور، فشار سیال و عدد ثابت تناسب نشان می‌دهد [8]. نرخ تخلیه بر اساس رابطه (1) محاسبه می‌شود [9]

$$q = \frac{-k\nabla P}{\mu L} \qquad (1)$$

که در آن، q میزان تخلیه، k میزان نفوذپذیری، μ میزان چسبندگی مایع، ∇P اختلاف فشار بین دو بخش و L طول بخش مربوط به تغییر فشار است. رابطه دارسی برای تخلیه بین نفرون و افرنت[9] در برخی عملگرها مورد استفاده قرار می‌گیرد. واژه‌های معادل مورد استفاده در الگوریتم NOA-2 در مقایسه با نفرون طبیعی در جدول (1) نمایش داده شده است.

جدول (1): واژه‌های معادل الگوریتم NOA_2 و نفرون طبیعی

نفرون طبیعی	الگوریتم NOA-2
پلاسمای درون نفرون و افرنت	جواب موجه
معیار غربالگری ذرات	تابع برازندگی
جریان ذرات بین افرنت و نفرون	جستجوی محلی
دفع ذرات مضر و جذب ذرات مفید	جستجوی سراسری
تفکیک ذرات بزرگ و کوچک	عملگر جداسازی
تغییر وضعیت ذرات در نفرون	عملگر میتوکندری
عملیات جذب و تراوش	عملگرهای جذب و تراوش
میزان نفوذپذیری و چسبندگی مایع	پارامترهای کنترلی k و μ
جهت‌گیری ذرات در نفرون	پارامتر کنترلی ρ
نرخ فیلتر ذرات در کپسول بومن	پارامتر کنترلی α %

در ادامه، ابتدا در بخش دوم، به معرفی پنج الگوریتم فراابتکاری که در این مقاله مورد مقایسه قرار گرفته‌اند، پرداخته شده و سپس در بخش سوم الگوریتم پیشنهادی شرح داده خواهد شد. در بخش چهارم کارایی الگوریتم پیشنهادی، مورد ارزیابی قرار گرفته و در نهایت در بخش پنجم به بررسی نتایج به دست آمده پرداخته می‌شود.

2. الگوریتم‌های فراابتکاری مشابه

در این بخش به معرفی مختصری از پنج الگوریتم فراابتکاری ژنتیک، رقابت استعماری، ازدحام ذرات، جغرافیای زیستی و اجتماع زنبور مصنوعی که در این مقاله جهت مقایسه نتایج با الگوریتم پیشنهادی NOA-2، استفاده شده‌اند، پرداخته می‌شود. همچنین نتایج الگوریتم پیشنهادی با نسخه اولیه آن مقایسه می‌گردد.

الگوریتم ژنتیک (GA[10]) از سیستم‌های بیولوژیکی الهام گرفته شده است. یک راه‌حل برای یک مساله در این الگوریتم به شکل رشته‌ای به نام «کروموزوم» نشان داده می‌شود که خود شامل

[1] Filtrating member
[2] Tubule
[3] Reabsorption
[4] Secretion
[5] Glomerulus
[6] Bowman's capsule
[7] Darcy's law
[8] Discharge
[9] Efferent
[10] Genetic Algorithm

مجموعه‌ای از عناصر به نام «ژن» است که مجموعه‌ای از مقادیر را برای متغیرهای بهینه‌سازی نگه می‌دارد [10]. الگوریتم ژنتیک با جمعیت تصادفی کروموزوم‌ها کار کرده و میزان تناسب هر کروموزوم با ارزیابی آن توسط یک تابع هدف تعیین می‌شود. برای شبیه‌سازی بقای طبیعی مناسب‌ترین فرآیند، بهترین کروموزوم‌ها را از طریق تقاطع و جهش برای تولید نسل بعد یا همان کروموزوم‌های فرزند مبادله می‌کنند. راه‌حل‌های فرزندان در صورتی که راه‌حل‌های بهتری نسبت به اعضای ضعیف جمعیت ارائه کنند، ارزیابی و برای تکامل جمعیت مورد استفاده قرار می‌گیرند. به طور معمول این فرآیند برای تعداد زیادی از نسل‌ها ادامه می‌یابد تا بهترین راه‌حل (نزدیک به بهینه) به دست آید.

در الگوریتم رقابت استعماری (ICA[1]) هر فرد از جمعیت را یک کشور می‌نامند و جمعیت به دو گروه مستعمرات و دولت امپریالیستی تقسیم می‌شود. رقابت بین امپریالیست‌ها برای تصاحب مستعمرات یکدیگر هسته اصلی این الگوریتم را تشکیل می‌دهد و در نهایت منجر به همگرایی کشورها می‌شود. در این رقابت امپراتوری‌های ضعیف به تدریج فرو می‌ریزند و تنها یک امپریالیست باقی می‌ماند که بقیه کشورها مستعمره آن خواهند بود [11].

الگوریتم بهینه‌سازی ازدحام ذرات (PSO[2]) از رفتار اجتماعی دسته‌ای از پرندگان مهاجر الهام گرفته شده است که تلاش می‌کنند به مقصدی ناشناخته برسند. در این الگوریتم، هر عضو یک «پرنده» در دسته است که «ذره» نامیده می‌شود. یک ذره مشابه کروموزوم (یا عضو جمعیت) در الگوریتم ژنتیک است، اما برخلاف الگوریتم ژنتیک، فرآیند تکاملی در بهینه‌سازی ازدحام ذرات، پرندگان جدیدی از پرندگان والدین ایجاد نمی‌کند. در عوض، پرندگان در جمعیت فقط رفتار اجتماعی و در نتیجه حرکت خود را به سمت مقصد تکامل می‌دهند [12].

الگوریتم بهینه‌سازی جغرافیایی زیستی (BBO[3])، یک روش جدید در حوزه بهینه‌سازی جهانی بر اساس نظریه جغرافیای زیستی است که نشان‌دهنده مطالعه توزیع جغرافیایی موجودات زیستی می‌باشد. متفاوت از سایر الگوریتم‌های مبتنی بر جمعیت، در الگوریتم بهینه‌سازی جغرافیایی زیستی، راه‌حل‌های ضعیف می‌توانند کیفیت‌ها را با پذیرش ویژگی‌های جدید از موارد خوب بهبود بخشند. این الگوریتم از طریق شبیه‌سازی مهاجرت گونه‌ها بین زیستگاه‌ها در فضای راه‌حل چندبعدی، که در آن هر زیستگاه یک راه‌حل کاندید را نشان می‌دهد، توسعه یافته است. مانند سایر الگوریتم‌های تکاملی، این الگوریتم به طور احتمالی اطلاعات را بین راه‌حل‌های کاندید به اشتراک می‌گذارد. جغرافیای زیستی نه‌تنها توصیفی از پراکندگی گونه‌ها، بلکه توضیحی جغرافیایی نیز ارائه می‌دهد. جغرافیای زیستی برحسب عواملی مانند مساحت زیستگاه و نرخ مهاجرت مدل‌سازی شده و تکامل، انقراض و مهاجرت گونه‌ها را توصیف می‌کند [13].

در الگوریتم اجتماع زنبور مصنوعی (ABC[4])، زنبورها شامل سه گروه کارگران، تماشاگران و پیشاهنگان هستند. در این الگوریتم، نیمه اول حل مساله شامل زنبورهای کارگر است و نیمه دوم را تماشاگران تشکیل می‌دهند. برای هر منبع غذایی، تنها یک زنبور شاغل وجود دارد. هر چرخه جستجو شامل سه مرحله است که عبارتند از فرستادن زنبورهای استخدام شده به منابع غذایی و سپس اندازه‌گیری مقدار شهد آنها، انتخاب منابع غذایی توسط تماشاگران پس از به اشتراک گذاشتن اطلاعات زنبورهای شاغل و در نهایت تعیین مقدار شهد غذاها. از این رو، رقص زنبورهای شاغل که شهد بالاتری دارند، تماشاگران را به مناطق غذایی با مقدار شهد بالاتر جذب می‌کند. زنبورها پس از رسیدن به منطقه انتخاب شده، بسته به اطلاعات بصری، منبع غذایی جدیدی را در همسایگی منبع موجود در حافظه انتخاب می‌کنند. اطلاعات بصری بر اساس مقایسه موقعیت‌های منبع غذایی است. هنگامی که شهد یک منبع غذایی توسط زنبورها رها می‌شود، یک منبع غذایی جدید به طور تصادفی توسط یک زنبور پیشاهنگ تعیین می‌شود و با منبع رها شده جایگزین می‌شود. در این مدل، در هر چرخه حداکثر یک پیشاهنگ برای جستجوی منبع غذایی جدید و تعداد شاغلان به بیرون می‌رود [14].

[1] Imperialist Competitive Algorithm
[2] Particle Swarm Optimization
[3] Biogeography-based Optimization
[4] Artificial Bee Colony optimization

علاوه بر الگوریتم‌های مطرح شده فوق، مطالعات دیگری در خصوص بهبود الگوریتم‌ها و یا ترکیب آنها برای ایجاد یک الگوریتم مناسب‌تر انجام شده است. از جمله مرجع [15] که با استفاده از ترکیب الگوریتم کرم شب‌تاب، الگوریتم ژنتیک و جستجوی محلی، یک الگوریتم ترکیبی برای کاهش خطا ارائه نموده است. در مرجع [16]، یک الگوریتم فرااکتشافی مبتنی بر رفتار پرنده تیهو ارائه شده است که با مقایسه با سایر الگوریتم‌ها خطای برون‌خطی را تا حدود بسیار زیادی کاهش می‌دهد. در مرجع [17]، الگوریتم اجتماع ذرات با روش‌های موازی‌سازی بهبود داده شده و برای حل مساله‌های بهینه‌سازی با محاسبات سنگین مورد استفاده قرار گرفته است.

3. الگوریتم نفرون-2

الگوریتم NOA-2 بر اساس نسخه قبلی این الگوریتم در کار پیشین نویسندگان [6]، توسعه داده شده است. نسخه پیشین با عنوان الگوریتم نفرون اولیه (NAO[1])، در سال 2016 توسط نویسندگان این مقاله ارائه گردید که پس از جداسازی با عملگر جهش در برخی از جواب‌های اولیه به تولید جواب‌های جدیدتر پرداخته و پس از آن با شبیه‌سازی عملگرهای جذب و تراوش نفرون طبیعی مکانیزم تمرکزگرایی را پیاده‌سازی می‌کند. در NOA-2، برخی از معادلات برای بهبود نتایج توسعه داده شده‌اند. همچنین عملگر جدیدی با عنوان میتوکندری به منظور ارتقا عملکرد الگوریتم در مکانیزم تنوع بخشی به الگوریتم اضافه شده است.

مراحل اجرای الگوریتم NOA-2 به صورت زیر می‌باشد:

1. مقداردهی اولیه: در این مرحله، پارامترهای الگوریتم تعیین می‌گردند. قبل از شروع اجرای هر الگوریتم بهینه‌سازی، ابتدا بایستی پارامترهای مهم در الگوریتم تنظیم شوند. پارامترهای الگوریتم NOA-2 شامل ضریب نفوذپذیری افرنت (K_{ef})، چسبندگی پلاسمای درون افرنت (μ_{ef})، ضریب نفوذپذیری نفرون (K_{nep}) و چسبندگی پلاسمای درون نفرون (μ_{nep}) است. همچنین α نرخ فیلتر برای پذیرش پلاسما درون نفرون و ρ پارامتر کنترلی برای تغییرات در میتوکندری است. حداکثر تکرار الگوریتم با $MaxIt$ نمایش داده می‌شود. این پارامترها در این مرحله مقداردهی اولیه می‌شوند.
2. ارزیابی جواب‌ها: هر پلاسما یا جواب با استفاده از تابع برازندگی مورد ارزیابی قرار می‌گیرد و بهترین جواب ذخیره می‌شود.
3. اجرای عملگرها: عملگرهای الگوریتم که شامل موارد زیر هستند تا رسیدن به شرط توقف یعنی بیشترین تکرار اجرا می‌شود.
   - جداساز: فاصله بین هر پلاسما با بهترین پلاسما محاسبه می‌گردد و به تعداد مشخصی بر اساس پارامتر نرخ فیلتر و پلاسماهایی که بیشترین فاصله را با بهترین پلاسما دارند، به بخش نفرون وارد می‌شوند و مابقی در بخش افرنت باقی می‌مانند.
   - میتوکندری: وضعیت پلاسماهای درون نفرون تغییر می‌کند و منجر به تولید جواب‌های متنوع می‌گردد.
   - جذب و تراوش: وضعیت پلاسماهای درون افرنت تغییر می‌یابد و منجر به جواب‌های متمرکز می‌شود.
   - دفع: تمامی پلاسماهای تغییر وضعیت یافته به پلاسماهای قبلی اضافه شده و بدترین‌های آنها حذف می‌گردند.

در ادامه این بخش، نحوه اجرای این عملگرها با جزئیات بیشتر شرح داده می‌شود.

جداساز: در عملگر اول با عنوان جداساز، درصدی از جمعیت برای ورود به نفرون بر اساس روابط (2) و (3) فیلتر می‌شوند. بر اساس این قانون، پلاسماهایی که ساختار آنها کمتر شبیه به بهترین پلاسما می‌باشد به نفرون وارد می‌شوند. معیار تعیین شباهت فاصله اقلیدسی است که در رابطه (2) ذکر شده است.

$$Dist_i(plasma_i, plasma_{best}) = \sqrt{\sum_{j=1}^{d}(plasma_{ij} - plasma_{best,j})^2} \quad (2)$$

1- Nephron Algorithm Optimization

$$N_{nep} = \alpha \times N_{plasma},\ N_{ef} = (1-\alpha) \times N_{plasma} \quad (3)$$

در این روابط N_{nep} تعداد پلاسمای درون نفرون و N_{ef} تعداد پلاسمای درون افرنت و N_{Plasma} تعداد کل پلاسمای موجود است.

میتوکندری: در عملگر دوم یا میتوکندری، برخی پلاسماهای درون نفرون طبق رابطه (4) تغییر موقعیت پیدا می‌کنند.

$$Mplasma_i = Mtch(plasma_i^{nep}) = \begin{cases} plasma_i + rand()plasma_i^{nep} & if\ P \le \rho \\ plasma_i - rand()plasma_i^{nep} & if\ P > \rho \end{cases} \quad (4)$$

یک پلاسما $plasma_i^{nep}$ به شکل تصادفی درون نفرون انتخاب شده و با ضریب تصادفی $rand()$ به موقعیت پلاسمای دیگر افزوده یا از آن کاسته می‌شود که موجب تغییر موقعیت آن می‌گردد.

جذب و تراوش: در عملگر سوم، برخی از پلاسماهای درون نفرون و افرنت طبق قانون جذب و تراوش موجود در رابطه‌های (5) و (6) تغییر موقعیت می‌دهند.

$$Rplasma_i = Reabsorption\,(plasma_i) = \begin{cases} plasma_i^{nep} + \beta\nabla d_{wi} + q_{ef} & \forall i \in Nep \\ plasma_i^{ef} + \beta\nabla d_{bi} + q_{ef} & \forall i \in Ef \end{cases} \quad (5)$$

$$Splasma_i = Secretion\,(plasma_i) = \begin{cases} plasma_i^{nep} + \frac{1}{\beta}\nabla d_{wi} + q_{nep} & \forall i \in Nep \\ plasma_i^{ef} + \frac{1}{\beta}\nabla d_{bi} + q_{nep} & \forall i \in Ef \end{cases} \quad (6)$$

پلاسمای درون افرنت با $plasma_i^{ef}$ نمایش داده شده و پارامتر β نیز طبق رابطه (7) محاسبه می‌شود:

$$\beta = \frac{N_{nep}}{N_{ef}} \quad (7)$$

رابطه‌های جذب و تراوش از سه بخش تشکیل شده‌اند که شامل (1) موقعیت کنونی پلاسما، (2) قانون تخلیه و (3) قانون جابجایی هستند. میزان تخلیه از نفرون به افرنت و بالعکس در رابطه‌های (8) و (9) نشان داده شده است. که در این روابط نمادهای $Avg(ef)$ و $Avg(nep)$ معرف میانگین موقعیت پلاسمای درون افرنت و نفرون می‌باشند.

$$q_{ef} = \frac{K_{ef}}{\mu_{nep}}[Avg(ef) - Avg(nep)] \quad (8)$$

$$q_{nep} = \frac{K_{nep}}{\mu_{ef}}[Avg(nep) - Avg(ef)] \quad (9)$$

در قانون جابجایی، طبق رابطه‌های (10) و (11) پلاسماهای نفرون به سمت بدترین پلاسما و پلاسماهای افرنت به سمت بهترین پلاسما تغییر موقعیت می‌دهند.

$$\nabla d_{wi} = (plasma_{worst} - plasma_i^{nep}) \quad (10)$$

$$\nabla d_{bi} = (plasma_{best} - plasma_i^{ef}) \quad (11)$$

دفع: در عملگر چهارم یا دفع، پلاسماهای زائد یا بدترین بر اساس نرخ جمعیت پلاسما حذف می‌شوند و مابقی در سیستم برای تکرار الگوریتم باقی می‌مانند. این عملگر در رابطه (12) نمایش داده شده است.

$$newPlasma = \text{SelectBetter}\{Mplasma_i \cup Rplasma_i \cup Splasma_i \cup plasma_{best}\} \quad (12)$$

3,1. بهبود در الگوریتم NOA-2 نسبت به نسخه اولیه

همانگونه که پیش از این اشاره شده، الگوریتم NOA-2 توسعه کاربردی‌تری از نسخه اولیه ارائه شده توسط نویسندگان است [6]. در این نسخه جدید، رابطها در اجرای عملگرها بهبود پیدا کرده و عملگری جدید با عنوان میتوکندری به این مرحله اضافه شده است. همان‌گونه که در سایر الگوریتم‌های فراابتکاری مشخص است، دو مولفه اصلی شامل تمرکزگرایی و تنوع‌بخشی وجود دارد که به جستجوی محلی و سراسری کمک می‌کنند. تبادل بین این دو مولفه کلیدی به صورت معنی‌داری بر کارایی الگوریتم تاثیر دارد. این دو مولفه در الگوریتم NOA-2 نیز نقش بسزایی دارند که در عملگرهای شرح داده شده در بالا کاملا مشهود است. در مرحله جداسازی، جواب‌هایی که به بهترین جواب شباهتی ندارند، به منظور تنوع‌بخشی جواب‌ها در همان ابتدای الگوریتم انتخاب می‌شوند. واضح است که با بکارگیری عملگر میتوکندری به این مقصود نزدیک‌تر می‌شویم زیرا تغییر موقعیت‌های تصادفی برای پلاسماها ایجاد شده و به جستجوی

سراسری کمک می‌شود. همانگونه که در ابتدای معرفی الگوریتم بیان شد، یک جهش ساده در تغییر موقعیت پلاسمای درون نفرون پیشین ایجاد می‌شد که منجر به تولید جواب‌های جدید می‌گردید، اما در الگوریتم پیشنهادی، عملگر جهش با شبیه‌سازی رفتار میتوکندری بر اساس رابطه (4) جایگزین گردید که در آن موقعیت پلاسمای درون نفرون به صورت تصادفی و مبتنی بر تصمیم احتمالی تغییر می‌یابد. با توجه به اینکه در این رابطه، نه تنها تغییرات موقعیت پلاسما به صورت تصادفی و با بکارگیری عدد تصادفی $rand$ انجام می‌شود، بلکه چگونگی تغییرات موقعیت هر پلاسما بر اساس تصمیم‌گیری احتمالی ρ می‌باشد، انتظار می‌رود الگوریتم در تکرارهای بعدی، جواب‌های متنوع‌تری تولید کند و دچار تله‌های محلی یا همگرایی زودرس نشود و با بهبود در تعادل بین مکانیزم‌های تنوع‌بخشی و تمرکزگرایی به جواب‌های بهتری دست یابد. از طرف دیگر، در مراحل جذب و تراوش، قانون تخلیه و قانون جابجایی به طور ترکیبی و بر اساس نرخ پارامترهای کنترلی موجود در عملگرها به تنوع‌بخشی و تمرکزگرایی کمک می‌کنند. هرچه نرخ فیلتر در عملگر جداسازی بیشتر باشد، تنوع‌بخشی افزایش می‌یابد و هر چه کمتر باشد تمرکزگرایی و همگرایی زودرس بیشتر می‌شود. همچنین ترکیب پارامترهای نفوذپذیری و چسبندگی در تعیین موقعیت‌های پلاسما موثر می‌باشند، به طوری که هرچه $\beta\nabla d_{wi}$، q_{ef}، $\beta\nabla d_{bi}$، $\frac{1}{\beta}\nabla d_{wi}$، $\frac{1}{\beta}\nabla d_{bi}$ و q_{nep} بیشتر شوند تنوع-بخشی افزایش می‌یابد و هرچه کمتر باشند تمرکزگرایی بیشتر خواهد شد. در الگوریتم (1) شبه کد اجرای الگوریتم NOA-2 پیشنهادی، ارائه شده است.

4. ارزیابی کارآیی الگوریتم

برای ارزیابی الگوریتم NOA-2، علاوه بر نسخه اولیه الگوریتم نفرون، پنج الگوریتم بهینه‌سازی فراابتکاری شامل الگوریتم ژنتیک، الگوریتم رقابت استعماری، بهینه‌سازی ازدحام ذرات، بهینه‌سازی جغرافیای زیستی و اجتماع زنبور مصنوعی بر روی هفت مساله بهینه‌سازی معروف اجرا شدند که نتایج و خروجی الگوریتم‌های فوق در این بخش ارائه می‌شود. تمامی این الگوریتم‌ها در نرم‌افزار متلب کدنویسی و در سیستمی با CPU، 2 گیگاهرتز و RAM، 1 گیگابایت اجرا شدند.

الگوریتم (1): شبه کد اجرای الگوریتم NOA-2

Algorithm. Nephron Optimization Algorithm-II

// Initialization

1. Parameter setting: $\mu_{ef}, \mu_{nep}, K_{ef}, K_{nep}, \alpha, \rho, N_{Plasma}, MaxIt$
2. Generate plasma solution: $plasma_i = \{1,2,\dots,d\} \in PlasmaPop$ randomly

// Evaluation

3. Evaluate plasma solutions and record the best plasma
4. **For** It **to** $MaxIt$ **do**

// Filtration

5. $P^{Nep} \coloneqq \emptyset,\ P^{Ef} \coloneqq PlasmaPop$
6. Calculate distance between the population and the best plasma based on Eqs. (2) and (3)
7. $SP = Sort(plasma_i, Descending\ Dist_i)$
8. **For** i **to** N_{nep} **do**
9. $P^{Nep}=P^{Nep} \cup \{SP_i\},\ P^{Ef}=P^{Ef} \setminus \{SP_i\}$
10. **End for**

// Mitochondria

11. Displace $plasma_i \in P^{Nep}$ according to the concentration gradient rule in Eq. (4)
12. Record each displaced plasma as $Mplasma_i^{nep} = Mitochodria(plasma_i^{nep})$

// Reabsorption

13. Displace $plasma_i \in P^{Nep}$ and $plasma_i \in P^{Ef}$ according to Eq. (5) and Eqs. (7), (8), (10), and (11)
14. Record displaced plasmas as $Rplasma$

// Secretion

15. Displace $plasma_i \in P^{Nep}$ and $plasma_i \in P^{Ef}$ according to Eq. (6) and Eqs. (7), (9), (10), and (11)
16. Record displaced plasmas as $Splasma$

// Excretion

17. Append displaced $plasmas$ to a new set solution according to Eq. (12)
18. Record the best $plasma$ found so far
19. **End for**
20. **Return** the best solution after meeting the stopping criteria

مساله‌های مورد بررسی شامل Ackley، Griewang، Rastrigin، Sphere، Rotated Hyper، Rosen Brock و Coolville می‌باشند که همگی از مساله‌های معروف و شناخته شده در بهینه‌سازی هستند و جزئیات آنها شامل توابع هدف، دامنه یا حدود متغیرها، جواب بهینه و ابعاد مساله یا تعداد متغیرها در جدول (2) نمایش داده شده است [17]. در ستون اول این جدول، نام هر مساله و تعداد متغیرها، در ستون دوم رابطه تابع هدف برای هر مساله که

به عنوان تابع برازندگی در الگوریتم‌های فراابتکاری فرموله می‌شوند، در ستون سوم مقدار بهینه تابع (جواب بهینه) که معیار ارزیابی کارآیی الگوریتم‌ها با توجه به محدوده متغیرها است که در ستون آخر نمایش داده شده است. هر الگوریتم 30 مرتبه بر روی هر مساله اجرا شده و میانگین و انحراف معیار جواب‌های به دست آمده الگوریتم NOA-2 با سایر الگوریتم‌ها مقایسه گردیده است. نتایج مقدار تابع هدف (جواب) و زمان محاسباتی الگوریتم برای یافتن جواب بهینه به ترتیب در جداول (3) و (4) نشان داده شده‌اند.

جدول (2): جزئیات مساله‌های معروف بهینه‌سازی مشتمل بر ابعاد مساله یا تعداد متغیرها، تابع هدف، جواب بهینه، و دامنه متغیرهای پیوسته

نام (اندازه)	محدوده x	$f(x^*)$	رابطه بهینه سازی
Ackley (128)	$[-32.768,32.768]$	0	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$
Griewangk (10)	$[-600,600]$	0	$f(x) = \sum_{i=1}^{d}\frac{x_i^2}{4000} - \prod_{i=1}^{d}\cos(\frac{x_i}{\sqrt{i}}) + 1$
Rastrigin (256)	$[-5.12,5.12]$	0	$f(x) = 10d + \sum_{i=1}^{d}[x_i^2 - 10\cos(2\pi x_i)]$
Sphere (256)	$[-5.12,5.12]$	0	$f(x) = \sum_{i=1}^{d}x_i^2$
Rotated-Hyper (256)	$[-65.536,65.536]$	0	$f(x) = \sum_{i=1}^{d}\sum_{j=1}^{i}x_j^2$
Rosenbrock (6)	$[-2.048,2.048]$	0	$f(x) = \sum_{i=1}^{d-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Colville (4)	$[-10,10]$	0	$f(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$

جدول (3): میانگین و انحراف معیار تابع هدف (جواب) برای هفت مساله بهینه‌سازی با اجرای شش الگوریتم و مقایسه با الگوریتم NOA-2

Problem	Criteria	GA	ICA	PSO	BBO	ABC	NAO	NOA-2
Ackley	AVG.	0.0685 *	0.574 *	2.0883 *	3.5246 *	15.159 *	8.9E-16	8.9E-16
	STDEV.	0.0054	0.150	0.4922	0.3326	0.6466	1E-31	1E-31
Griewangk	AVG.	0.036 *	0.048 *	0.0607 *	0.0463 *	0.0556 *	0	0
	STDEV.	0.018	0.014	0.02257	0.02076	0.01654	0	0
Rastrigin	AVG.	0.296 *	8.830 *	263.63 *	418.77 *	1931*	1.60E-7*	0
	STDEV.	0.051	1.554	28.1575	28.149	167	3.424E-7	0
Sphere	AVG.	0.44 *	7.4E-06*	0.1247 *	2.19 *	0.039 *	3.8E-62	0
	STDEV.	0.08	1.2E-06	0.0723	0.18	0.013	1.6E-61	0
Rotated Hyper	AVG.	83.536 *	0.0007 *	1.2057 *	422.77*	3.9839 *	1.1E-56	0
	STDEV.	12.931	0.0005	0.2953	89.31	1.49265	3.9E-56	0
Rosen Brock	AVG.	1.958 *	0.0003*	7.1E-05 *	0.1279 *	0.1257 *	0.206*	8.42E-17
	STDEV.	0.353	0.0005	8.4E-05	0.08308	0.05642	0.128	4.53E-16
Colville	AVG.	0.009 *	0.0009 *	7.58E-09*	0.00142*	0.0010 *	3.44E-6*	4.2E-28
	STDEV.	0.007	0.0006	1.23E-08	0.00302	0.00021	4.59E-6	2.3E-27

* نشان می‌دهد که NOA-2 در سطح معنی‌داری 0/05 برای حل مساله مورد نظر، از لحاظ کیفیت جواب دارای عملکرد بهتری نسبت به سایر الگوریتم‌های بهینه‌سازی می‌باشد.

جدول (4): میانگین و انحراف معیار زمان محاسباتی برای هفت مساله بهینه‌سازی با اجرای شش الگوریتم و مقایسه با الگوریتم NOA-2

Problem	Criteria	GA	ICA	PSO	BBO	ABC	NAO	NOA-2
Ackley	AVG.	49.96 *	58.45 *	45.5 *	63.282 *	18.9 *	7.68*	3.33
	STDEV.	1.483	0.049	2.864	1.257	1.4	0.365	0.31
Griewangk	AVG.	4.94 *	9.726 *	1.06 *	6.9798 *	25.13 *	7.43*	0.26
	STDEV.	0.54	0.273	0.16	0.0845	4.426	1.12	0.04
Rastrigin	AVG.	127.6 *	119.82 *	58.12 *	354.97 *	174.3*	2.73*	0.55
	STDEV.	0.1304	1.1909	0.188	15.841	0.289	0.29	0.058
Sphere	AVG.	40.6 *	38.19 *	54.07 *	232.02 *	98.097 *	7.76*	4.99
	STDEV.	3.064	0.528	0.125	3.7983	1.3915	0.31	0.14
Rotated Hyper	AVG.	542.2 *	216.58 *	622.6 *	375.6*	1190 *	77.58*	16.27
	STDEV.	2.608	1.4802	2.872	15.6	1.818	2.09	1.061
Rosen Brock	AVG.	185.1 *	73.41 *	38.49 *	513.13 *	145.79 *	47.35*	32.13
	STDEV.	9.674	1.011	1.47	1.7061	2.5206	1.07	0.815
Colville	AVG.	13.27 *	66.77 *	36.48 *	73.7 *	40.38 *	7.26*	3.55
	STDEV.	1.343	1.06	2.05	0.899	1.566	0.477	0.306

* نشان می‌دهد که NOA-2 در سطح معنی‌داری 0/05 برای حل مساله مورد نظر، از لحاظ زمان محاسباتی دارای عملکرد بهتری نسبت به سایر الگوریتم‌های بهینه‌سازی می‌باشد.

همانگونه که در جدول (3) مشاهده می‌شود، الگوریتم NOA-2 در مقایسه با شش الگوریتم دیگر جواب بهینه یا نزدیک بهینه را در زمان معینی یافته است و نتایج بهتری ارائه داده است. زمینه خاکستری تیره و روشن به ترتیب الگورتیم رتبه اول و دوم را از نظر نتایج (کیفیت جواب) نشان می‌دهد و نماد * نشان می‌دهد الگوریتم نفرون در مقایسه با الگوریتم‌های دارای نماد به صورت معناداری (در سطح معنی‌داری 0/05) جواب بهتری به دست آورده است. مقدار p-value بر اساس آزمون Dunnett T3 محاسبه گردیده و برای غالب روش‌های بهینه‌سازی در مقایسه با الگوریتم پیشنهادی کمتر از سطح معنی داری 0/05 به دست آمده است که این امر گویای این است که کیفیت جواب‌های سایر الگوریتم‌ها با کیفیت جواب‌های الگوریتم پیشنهادی تفاوت معنی‌داری دارد.

جدول (4) نتایج تلاش و زمان محاسباتی شش الگوریتم مورد بررسی را در مقابل NOA-2 نمایش می‌دهد. در این جدول نیز مقدار p-value بر اساس آزمون Dunnett T3 محاسبه گردیده و برای کلیه روش‌های بهینه‌سازی در مقایسه با الگوریتم پیشنهادی کمتر از سطح معنی‌داری 0/05 به دست آمد و گویای این است که زمان محاسباتی سایر الگوریتم‌ها برای حل مساله‌های بهینه-سازی با زمان محاسباتی الگوریتم پیشنهادی تفاوت معنی‌داری دارد. الگوریتم پیشنهادی NOA-2 به طور معناداری می‌تواند در زمان کمتری نسبت به سایر الگوریتم‌ها جواب بهینه مساله‌های بهینه‌سازی آزمون را پیدا کند.

5. بحث و نتیجه‌گیری

از نتایج به دست آمده در جداول (3) و (4)، مشاهده می‌شود که الگوریتم NOA-2 در مقایسه با الگوریتم‌های مورد بررسی توانسته است بهترین جواب را در کمترین زمان محاسباتی پیدا کند. عملگر میتوکندری به الگوریتم کمک می‌کند تا جواب بهینه مساله‌ها سریع‌تر از الگوریتم‌های دیگر به دست بیاید زیرا قدرت تنوع‌بخشی الگوریتم را افزایش می‌دهد. از طرف دیگر، دو عملگر جذب و تراوش نقش کلیدی در ایجاد توازن بین مکانیزم‌های تنوع‌بخشی و تمرکزگرایی الگوریتم را ایفا می‌کنند.

در ضمن، نتایج نشان می‌دهد انحراف معیار جواب‌های الگوریتم NOA-2 بسیار کم بوده و این موضوع پایایی الگوریتم را در بین سایر الگوریتم‌های مورد بررسی، تایید می‌کند. شایان ذکر است که الگوریتم پیشنهادی در مقایسه با پنج الگوریتم معروف بهینه‌سازی از دو جنبه کیفیت جواب و زمان محاسباتی به طور معناداری عملکرد بهتری دارد. همچنین از جنبه زمان محاسباتی در حل کلیه مسائل بهینه‌سازی به طور معناداری نسبت به نسخه اولیه الگوریتم نفرون دارای قابلیت بالایی است و از طرف دیگر در حل سه مساله Rastrigin، Rosen Brock و Coolville به طور معناداری نسبت به نسخه اولیه به جواب‌های بهتری دست یافته است. این نتایج گویای برتری الگوریتم توسعه یافته نفرون نسبت به نسخه قبلی آن در یافتن جواب بهتر با سرعت بیشتر می‌باشد. دقت کنید که اگرچه تفاوت معناداری بین جواب‌های الگوریتم پیشنهادی و نسخه اولیه آن در حل مساله‌های Ackley، Griewangk، Sphere و Rotated Hyper مشاهده نمی‌شود و کیفیت جواب‌های مساله‌های Ackley و Griewangk در دو الگوریتم نفرون توسعه یافته و نسخه پیشین یکسان است، ولی با توجه به انحراف معیار به نسبت بالای جواب‌های یافته شده دو مساله Sphere و Rotated Hyper در نسخه اولیه نفرون، نتیجه می‌گیریم که پایایی الگوریتم جدید نسبت به نسخه پیشین آن نیز بهبود یافته است. با توجه به این که مساله‌های نمونه آزمون به صورت پیوسته بودند، لذا الگوریتم NOA-2 می‌تواند برای حل مسائل بهینه‌سازی پیوسته امیدبخش باشد. پیشنهادها برای پژوهش‌های آینده در این زمینه این است که الگوریتم NOA-2 برای حل مساله‌های گسسته و بهینه‌سازی ترکیباتی نظیر فروشنده دوره‌گرد، کوله‌پشتی و حوزه زمانبندی نیز مورد بررسی قرار گیرد.

تعارض منافع: نویسندگان اعلام می‌کنند که هیچ تعارض منافعی ندارند.

مراجع

[1] L. Bianchi, M. Dorigo, L.M. Gambardella, and W.J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," Nat. Comput., vol. 8, no. 2, pp. 239-287, 2009, doi: 10.1007/s11047-008-9098-4.

[2] M.G.H. Omran, S. Al-Sharhan, and M. Clerc, "A modified Intellects-Masses Optimizer for solving real-world optimization problems," Swarm Evol. Comput., vol. 41, pp. 159-166, 2018, doi: 10.1016/j.swevo.2018.02.015.

[3] M. Lindahl, M. Sorensen, and T.R. Stidsen, "A fix-and-optimize matheuristic for university timetabling," J. Heuristics, vol. 24, no. 4, pp. 645-665, 2018, doi: 10.1007/s10732-018-9371-3.

[4] C.-C. Wu, J.-Y. Chen, W.-C. Lin, K. Lai, S.-C. Liu, and P.-W. Yu, "A two-stage three-machine assembly flow shop scheduling with learning consideration to minimize the flowtime by six hybrids of particle swarm optimization," Swarm Evol. Comput., vol. 41, pp. 97-110, 2018, doi: 10.1016/j.swevo.2018.01.012.

[5] R. Capua, Y. Frota, L.S. Ochi, and T. Vidal, "A study on exponential-size neighborhoods for the bin packing problem with conflicts," J. Heuristics, vol. 24, no. 4, pp. 667-695, 2018, doi: 10.1007/s10732-018-9372-2.

[6] R. Behmanesh, "Nephron Algorithm Optimization: Inspired of the Biologic Nephron Performance," Int. J. Appl. Metaheuristic Comput., vol. 7, no. 1, pp. 38-64, 2016, doi: 10.4018/IJAMC.2016010103.

[7] A. Maton, J. Hopkins, C.W., McLaughlin, S. Johnson, M.Q. Warner, D. LaHart, and J.D. Wright, Human Biology and Health. Prentice Hall, Englewood Cliffs, 1993.

[8] H. Darcy, Les fontaines publiques de la ville de Dijon: exposition et application, par Henry Darcy. Victor Dalmont, 1856.

[9] A. Bejan, Convection heat transfer, John Wiley & Sons, Inc., 2013, doi: 10.1002/9781118671627.

[10] D.E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addison-Wesley 1989, ISBN 0-201-15767-5.

[11] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: An algorithm for optimization

inspired by imperialistic competition," in IEEE Congr. Evol. Comput., Singapore, 2007, pp. 4661-4667, doi: 10.1109/CEC.2007.4425083.

[12] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in IEEE Int. Conf. Evol. Comput. Proc., IEEE World Congr. Comput. Intell., Anchorage, AK, USA, 1998, pp. 69-73, doi: 10.1109/ICEC.1998.699146.

[13] D. Simon, "Biogeography-Based Optimization," IEEE Trans. Evol. Comput., vol. 12, no. 6, pp. 702-713, 2008, doi: 10.1109/TEVC.2008.919004.

[14] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," J. Glob. Optim., vol. 39, no. 3, pp. 459-471, 2007, doi: 10.1007/s10898-007-9149-x.

[15] J. Salimisartaghti and S. Goli-Bidgoli, "A Hybrid Algorithm using Firefly, Genetic, and Local Search Algorithms," Soft Comput. J., vol. 8, no. 1, pp. 14-28, 2019, doi: 10.22052/8.1.14 [In Persian].

[16] M. Mohammadpour, B. Minaei, and H. Parvin, "Introducing a new meta-heuristic algorithm based on See-See Partridge Chicks Optimization to solve dynamic optimization problems," Soft Comput. J., vol. 8, no. 2, pp. 38-65, 2020, doi: 10.22052/8.2.38 [In Persian].

[17] M.P. Akbarpour, K. Khamforoosh, and V. Maihami, "An approach to Improve Particle Swarm Optimization Algorithm Using CUDA," Soft Comput. J., vol. 8, no. 2, pp. 2-21, 2020, doi: 10.22052/8.2.2 [In Persian].

[18] S. Surjanovic and D. Bingham, D., (August 2017), Optimization Test Problems, Simon Fraser University, [Online], Available: https://www.sfu.ca/~ssurjano/optimization.html.