مجله علمی-پژوهشی
رایانش نرم و فناوری اطلاعات
جلد۲ ، شماره۱، سال ۱۳۹۲

الگوریتم بهینه‌سازی نیروی مرکزی چند هدفه

حامد نجف زاده[1]، سید حمید ظهیری[2]
[1] دانشگاه بیرجند، دانشکده برق و کامپیوتر، بیرجند، najafzadeh۶۶@yahoo.com
[2] دانشگاه بیرجند، دانشکده برق و کامپیوتر، بیرجند، shzahiri@yahoo.com

چکیده:
در این مقاله روش جدیدی در بهینه‌سازی چندهدفه مبتنی بر الگوریتم بهینه‌سازی نیروی مرکزی به نام الگوریتم بهینه سازی نیروی مرکزی چندهدفه **(MOCFO)** *ارائه می‌شود. روش* **MOCFO** *از مفهوم «بهینگی پرتو» برای شناسایی موقعیت‌های «غیرچیره شده» و از یک «مخزن بیرونی» برای نگهداری این موقعیت‌ها استفاده می‌کند. برای اطمینان از صحت عملکرد روش ارائه شده در مواجه با مسائل بهینه‌سازی چندهدفه، آن را برروی توابع استاندارد معتبر مورد آزمایش قرار می‌دهیم. نتایج نهایی قدرت و عملکرد این روش را نشان می‌دهد که بستر جدیدی از تحقیقات را فراروی محققین قرار داده است.*

کلید واژه‌ها: *بهینه‌سازی چندهدفه، الگوریتم بهینه‌سازی نیروی مرکزی، بهینگی پرتو.*

۱. مقدمه

حل مسائل مهندسی، به معنای رسیدن به بهترین پاسخ برای آن مسائل است که با صرف کم‌ترین هزینه و بالاترین سرعت به دست آمده باشد. لذا مهندسی را می‌توان به نوعی علم بهینه سازی مسائل دانست. بسیاری از مسائل بهینه‌سازی دارای چندین هدف هستند و در آنها باید به صورت همزمان چندین تابع هدف را بهینه نمود.

حل مسائل چند هدفه با روش‌های سنتی که دارای گرادیان هستند به دلیل بالا بودن ابعاد و همچنین تعداد زیاد ورودی و خروجی همزمان در این نوع مسائل عملا امکان‌پذیر نمی‌باشد. لذا با گذشت زمان، رشد اطلاعات علمی و افزایش ابعاد داده‌ها نیاز دو چندان برای حل مسائل چند هدفه که اکثراً در صنایع و کاربردهای عملی نمایان می‌شوند، موجب ابداع روش‌های نوین چند هدفه توسط محققان گردید. در چند سال اخیر الگوریتم های مختلفی برای حل این دسته از مسائل بهینه‌سازی چند هدفه در زمینه‌های مختلف به کارگرفته شد و نتایج آنها بررسی شد.

از جمله معروف‌ترین و موفق‌ترین این روش‌ها می‌توان روش‌های بهینه‌سازی چند هدفه مبتنی بر بهینه‌سازی پِرِتو را نام برد. در این روش‌ها یک مفهوم اساسی، سبب شده تا الگوریتم بتواند به پاسخ بهینه دست پیدا کند. این مفهوم "چیره بودن[1]" می‌باشد که به صورت زیر تعریف می‌گردد:

جواب x_i را بر x_j "چیره" گویند هرگاه دو شرط زیر در مورد آن محقق شود:

۱- جواب x_i از دید هیچ یک از توابع هدف از x_j بدتر نباشد.

۲- جواب x_i حداقل در یکی از توابع هدف از x_j بهتر باشد.

با توجه به این تعریف گفته می‌شود روند یافتن پاسخ بهینه در صورت داشتن چنین شرایطی به "بهینگی پِرتو[2]" منجر خواهد شد. $x_i \in U$ را "بهینه پِرتو" گویند اگر و تنها اگر در مجموعه مرجع جواب‌ها (U)، هیچ $x_j \in U$ وجود نداشته باشد که بر x_i "چیره" باشد و مجموعه جواب‌های

---
[1] Domination
[2] Pareto-Optimality

x_i را مجموعه "جبهه پرتو[۱]" می‌خوانند [۱].

در بین الگوریتم‌های ابتکاری برای حل مسائل چند هدفه ارائه شده تاکنون، می‌توان به الگوریتم‌های تکاملی چند هدفه NPGA[۲]، NSGA I & II[۳] و الگوریتم هوش جمعی MOPSO[۴] را نام برد [۲و۳].

الگوریتم CFO[۵] یکی از الگوریتم‌های جدید در زمینه بهینه‌سازی است که با الهام از قوانین گرانش فیزیک بین اجسام سعی در حل مسائل بهینه‌سازی را دارد [۴ و ۵]. در این مقاله یک روش جدید در بهینه‌سازی مسائل چند هدفه مبتنی بر الگوریتم بهینه‌سازی نیروی مرکزی به نام الگوریتم بهینه‌سازی نیروی مرکزی چند هدفه (MOCFO)[۶] معرفی می‌گردد. در روش پیشنهادی از بهینگی پَرِتو نقاط غیر چیره پیدا شده و آنها در یک حافظه جانبی (مخزن) ذخیره می‌گردند.

برای بررسی عملکرد روش MOCFO در حل مسائل بهینه‌سازی چند هدفه، نتایج آن بر روی یک سری توابع آزمون استاندارد، با دو روش معروف NSGA II و MOPSO مقایسه شد که نتایج در قالب نمودارها بیان شده است.

پیکربندی این مقاله به این صورت است که ابتدا در فصل ۲ بعد از شرح مختصری در مورد مفاهیم اساسی و اولیه روش‌های چند هدفه، در فصل ۳ الگوریتم ابتکاری CFO مختصراً توضیح داده خواهد شد. در فصل ۴ جزئیات روش ارائه شده بیان گشته و نتایج این روش بر روی توابع آزمون استاندارد و نتایج مقایسه‌ای در فصل ۵ بررسی شده است. نهایتاً نتیجه‌گیری در فصل ۶، پایان بخش مقاله خواهد بود.

۲. مفاهیم اولیه

تعاریف اولیه روش‌های بهینه‌سازی چند هدفه به طور قراردادی، به شرح زیر بیان می‌شوند [۶].

تعریف ۱: مسئله بهینه سازی چند هدفه (MOP)

یک MOP متداول شامل مجموعه‌ای از n پارامتر (متغیرهای تصمیم)، مجموعه‌ای از k تابع هدف، و مجموعه‌ای از m قید است. توابع هدف و قیود، توابعی از متغیرهای تصمیم هستند.

هدف از بهینه‌سازی این هست که:

$$\begin{aligned} &minimize \quad y=f(x)=(f_1(x),f_2(x),\ldots,f_k(x)) \qquad (۱)\\ &subjectto \;\; e(x)=(e_1(x),e_2(x),\ldots,e_m(x))\le 0\\ &where \quad x=(x_1,x_2,\ldots,x_n)\in X\\ &\qquad\qquad y=(y_1,y_2,\ldots,y_k)\in Y \end{aligned}$$

به طوری که x بردار تصمیم[۷] و y بردار هدف[۸] است، X به عنوان فضای تصمیم[۹] مشخص می‌شود و Y فضای هدف[۱۰] نامیده می‌شود. قیود e(x)≤۰ مجموعه‌ای از پاسخ‌های ممکن[۱۱] را تعیین می‌کند.

تعریف ۲: برای دو بردار $x,y\in R^k$، گفته می‌شود $x\le y$ اگر برای هر $i=1,\ldots,k$، $x_i\le y_i$، همچنین گفته می‌شود x بر y چیره است اگر $x\le y$ و $x\ne y$.

تعریف ۳: برای یک بردار از متغییرهای تصمیم $x\in X\subset R^n$ گفته می‌شود x نسبت به X غیر چیره شده[۱۲] است اگر $x'\in X$ دیگری وجود نداشته باشد به طوری که $f(x')\prec f(x)$.

تعریف ۴: برای یک بردار از متغییرهای تصمیم $x^*\in F\subset R^n$ (F مجموعه جواب‌های ممکن) بهینگی پَرِتو برقرار است اگر x^* در خصوص بردار F غیر چیره شده باشد.

تعریف ۵: مجموعه بهینه پَرِتو P^* به صورت زیر تعریف می‌گردد:

$$P^*=\{x\in F\mid x\ is\ Pareto-Optimal\}$$

تعریف ۶: جبهه پَرِتو PF^* به صورت زیر بیان می‌گردد:

$$PF^*=\{f(x)\in R^k\mid x\in P^*\}$$

۳. الگوریتم بهینه‌سازی نیروی مرکزی (CFO)

الگوریتم بهینه سازی نیروی مرکزی یکی از الگوریتم‌های ابتکاری و الهام گرفته از طبیعت می‌باشد که بر

---

[۱] Pareto-front

[۲] Niched Pareto Genetic Algorithm

[۳] Non-dominated Sorting Genetic Algorithm I and II

[۴] Multi Objective Particle Swarm Optimization

[۵] Central Force Optimization

[۶] Multi Objective Central Force Optimization

[۷] Decision vector

[۸] Objective vector

[۹] Decision space

[۱۰] Objective space

[۱۱] Feasible solutions

[۱۲] Nondominated

طبق علم مربوط به حرکت و نیروها، ساخته شده است. این علم شاخه‌ای از فیزیک می‌باشد که به بررسی حرکت اجرام تحت تاثیر نیروی گرانشی می‌پردازد. به دلیل این که معادلات حرکت به صورت قطعی[۱] هستند، لذا الگوریتم CFO از دیگر الگوریتم‌های بر گرفته از طبیعت متمایز شده و دارای خاصیت قطعیت در رسیدن به جواب می‌باشد و در طی رسیدن به جواب از متغیرهای تصادفی استفاده نمی‌کند [۴].

در ادامه به بررسی معادلات و نحوه عملکرد این الگوریتم می‌پردازیم. CFO توسط ریچارد فورماتو[۲] در سال ۲۰۰۶ نوشته شد که از این مفهوم برگرفته شده است که در کائنات تمامی اجرام توسط نیروی جاذبه به یکدیگر جذب می‌گردند که این امر بیان کننده قانون جهانی گرانش نیوتن می‌باشد. نیروی گرانش یک نیروی برداری بوده که بر روی اشیاء در فواصل دور اعمال می‌گردد و میزان این نیرو به دوری یا نزدیکی و میزان جرم‌های دو شئ بستگی دارد، حال برطبق فرمول این نیرو به حاصلضرب جرم دو جسم نسبت مستقیم و به فاصله بین مراکز دو جسم به صورت مجذوری نسبت عکس دارد به همین دلیل هم نام این الگوریتم بهینه‌سازی نیروی مرکزی است.

فضای تصمیم به صورت $x_i^{\min} \le x_i \le x_i^{\max}, i = 1, 2, \ldots, N_d$ تعریف شده است که در آن x_i متغیرهای تصمیم هستند که موقعیت نقطه بهینه را برای تابع هدف $f(x_1, x_2, \ldots, x_{N_d})$ تعیین می‌کنند. $f(\vec{x})$ در هر نقطه $\vec{x}$ برآزندگی (fitness) آن نقطه است. در اینجا هم مانند دیگر الگوریتم‌ها توپولوژی تابع هدف به صورت نامعلوم است.

در جهان فیزیک مقدار نیروی گرانشی بین دو جرم m_1 و m_2 که به فاصله r از هم قرار گرفته‌اند به صورت زیر به دست می‌آیند (Marion ۱۹۷۰) :

$$F = \gamma \frac{m_1 m_2}{r^2} \qquad (۲)$$

به طوری که $\gamma \succ 0$ یک ثابت گرانشی است. شتاب ناشی از نیروی اعمالی از جرم m_2 به جرم m_1 از قانون دوم نیوتن به صورت زیر حساب می‌گردد:

$$\vec{a}_1 = -\gamma \frac{m_2 \hat{r}}{r^2} \qquad (۳)$$

به طوری که $\hat{r}$ یک بردار است که از جرم m_2 خارج شده و در جهت m_1 در مسیر خط مستقیم بین دو جرم فوق قرار دارد.

با توجه به این معادله موقعیت ذره با گذشت زمان Δt به صورت زیر در سه بعد تغییر می‌کند:

$$\vec{R}(t+\Delta t) = \vec{R}_0 + \vec{V}_0 \Delta t + \frac{1}{2}\vec{a}\Delta t^2 \qquad (۴)$$

در این رابطه $\vec{R}(t+\Delta t)$ موقعیت ذره در زمان $t+\Delta t$ است و $\vec{R}_0$ و $\vec{V}_0$ به ترتیب بردارهای موقعیت و سرعت در زمان t هستند. موقعیت ذرات در بیشتر الگوریتم‌ها در یک فضای سه بعدی کارتزین می‌باشد : $\vec{R} = x\hat{i} + y\hat{j} + z\hat{k}$ که در آن $\hat{i}$، $\hat{j}$ و $\hat{k}$ بردارهای یکه در جهت محورهای مختصات هستند ولی در CFO در یک فضای بیشتر از سه بعد تا N_d بعد نیز فعالیت می‌کند تا بتوان هر تابع هدف با هر تعداد بعد را حل کند.

در الگوریتم فوق گروهی از ذرات به نام پراب (prob) در فضای تصمیم با استفاده از فرمول‌های حرکت جابه‌جا می‌شوند و موقعیت آنها جواب مسئله را در هر لحظه گسسته از زمان به ما نشان می‌دهد. $\vec{R}_j^p$ نشان‌دهنده پراب P ام در زمان J ام است. برای مثال پراب P از زمان J-۱ ($\vec{R}_{J-1}^p$) به زمان J ، ($\vec{R}_J^p$) منتقل می‌شود که این عمل در تمام ابعاد آن رخ می‌دهد و در حالت کلی هر پراب به به صورت زیر می‌باشد:

$$\vec{R}_j^p = \sum_{k=1}^{N_d} x_k^{p,j} \hat{e}_k \qquad (۵)$$

که در آن $x_k^{p,j}$ پراب p ام در زمان J ام و $\hat{e}_k$ بردار واحد در جهت x_k است. تابع هدف که در N_d بعد است، در زمان J-۱ برای پراب P به صورت زیر تعریف می‌شود:

$$M_{j-1}^p = f\left(x_1^{p,j-1}, x_2^{p,j-1}, \ldots, x_{N_d}^{p,j-1}\right) \qquad (۶)$$

در الگوریتم CFO برای هر پراب یک برآزندگی به صورت بالا تعریف می‌شود که $M_{j-1}^p, k = 1,2,\ldots,N_p$ تعداد کل پراب‌ها است. حرکت پراب از موقعیت $\vec{R}_{J-1}^p$ به $\vec{R}_J^p$ بستگی

---
[۱] Deterministic
[۲] Richard A.Formato

به مقادیر اولیه موقعیت و مقدار شتاب اعمالی بر دو پراب دارد که از رابطه زیر برای مثلا دو پراب p و n به دست می‌آید:

$$A = \frac{G.U\left(M_{j-1}^{n} - M_{j-1}^{p}\right)\left(M_{j-1}^{n} - M_{j-1}^{p}\right)^{\alpha}\left(\vec{R}_{j-1}^{n} - \vec{R}_{j-1}^{p}\right)}{\left|\left(\vec{R}_{j-1}^{n} - \vec{R}_{j-1}^{p}\right)\right|^{\beta}} \qquad (۷)$$

$G \succ 0$ ثابت گرانشی است که معادل است با مقدار γ در فرمول فیزیکی قانون گرانش، U تابع پله می‌باشد و α و β دو عدد ثابت با مقادیر مثبت می‌باشند.

طبق توضیحات فوق جستجو در فضای تصمیم بـه ایـن ترتیب انجام می‌پـذیرد کـه در ابتـدا یـک جمعیـت اولیـه از پروب‌هـا بـه صـورت تصـادفی ایجـاد و نسـبت بـه محاسـبه برازندگی آنها اقدام می‌گردد. سپس بـرای هـر پـروب مقـدار شتاب ناشی از نیروهای وارد بر آن بـا اسـتفاده از رابطـه (۷) محاسبه می‌گردد.

موقعیت جدید هر پراب را با استفاده از شتاب به دسـت آمده توسط رابطه (۸) به روز رسانی می‌شود:

$$R_j^p = R_{j-1}^p + \frac{1}{2} A_{j-1}^p \times \Delta t^2 \qquad (۸)$$

در رابطه فوق A_{J-1}^{p} شتاب وارد بر پروب p ام و R_{J-1}^{p} موقیت قبلی آن پروب می‌باشد. با توجه به این که عمل جستجو به صورت گسسته زمان صورت می‌گیرد، مقدار Δt برابر واحد در نظر گرفته می‌شود. ممکن است در این تغییر مکان، پروب‌ها از مرزهای فضای پاسخ خارج شوند که در این صورت هر یک از آنها را به وسیله عملگر خاصی (فاکتور تغییر موقعیت)[۱] به مقداری در داخل مرزهای فضای پاسخ تغییر مقدار داده می‌شود.

اکنون مجددا مقادیر برازندگی پروب‌ها را محاسبه و تغییر مکان آنها را پس از محاسبه شتاب وارد بر هر یک از آنها محاسبه و اِعمال می‌کنیم. این فرایند تا حصول به یکی از معیارهای توقف (که می‌تواند رسیدن به یک برازندگی مطلوب و یا تعداد تکرار پیش فرض باشد) ادامه می‌یابد [۴ و ۵].

به طور خلاصه روند رسیدن به پاسخ در الگوریتم فوق به صورت زیر است:

۱- موقعیت‌های اولیه پراب‌ها را حساب کرده و مقدار برازندگی را برای هر یک از آنها با استفاده از تابع هدف به دست می‌آوریم. سپس به هر پراب مقدار اولیه شتاب را با استفاده از فرمول شتاب اعمال می‌کنیم.

۲- موقعیت جدید هر پراب را با استفاده از فرمول شتاب و فرمول حرکت به روز رسانی می‌کنیم.

۳- تعیین می‌کنیم که آیا همه پراب‌ها در فضای تصمیم (Decision Space) قرار دارند.

۴- برآزندگی هر پراب را محاسبه می‌کنیم.

۵- به جز مرحله ۱ مراحل بالا را آنقدر تکرار کرده تا یکی از معیارهای توقف رخ دهد.

۴. معرفی روش MOCFO

در روش MOCFO از بهینه‌سازی پرتـو بـرای شناسـایی پاسخ های "غیر چیره شده" و از یک "مخزن بیرونی" بـرای نگهداری این پاسخ ها استفاده می‌شود.

در این الگوریتم ابتدا جمعیـت اولیـه ای از پـراب‌هـا بـه شـکل تصـادفی انتخـاب مـی‌شـود. در هـر مرحلـه پـس از محاسبه‌ی توابع برازندگی، بهترین پراب هـا در یـک مخـزن بیرونی شامل پاسخ‌های پرتو نگهداری می‌شود. سپس باتوجه به روابط الگوریتم CFO، موقعیت هر پراب برای تکرار بعدی الگوریتم بروز رسانی می‌شود.

الگوریتم MOCFO شامل مراحل زیر است:

۱- ایجاد جمعیت اولیه (POP).

در این مرحله یک جمعیت اولیه شامل تعداد معینی (Np) از پراب ها کاملا به طور تصادفی ایجاد می‌شود.

$$Initialize \quad POP_p^t, \quad t=0, \quad p=1,2,\cdots,N_p \qquad (۹)$$

۲- تعیین شتاب اولیه هر پراب.

در این مرحله شتاب اولیه صفر برای هر پراب در نظر گرفته می‌شود.

$$A_p^t = 0, \quad t=0, \quad p=1,2,\cdots,N_p \qquad (۱۰)$$

۳- ارزیابی هریک از پراب‌ها.

در این مرحله توابع برازندگی[۲] هر پراب محاسبه می‌شود.

---

[۱] Reposition Factor (F_{rep})

[۲] Fitness functions

$$M_p^t = f\left(POP_p^t\right), \quad t=0, \quad p=1,2,\cdots,N_p \qquad (۱۱)$$

۴- ذخیره سازی موقعیت پراب های غیرچیـره شـده در مخزن REP.

۵- تولید ابرمکعب‌هایی [۱] در مناطق مرورشده در فضای پاسخ و قرار دادن پراب های غیرچیره شده درون این ابرمکعب‌ها.

۶- ایجاد حلقه‌ی اصلی جستجو برای Nt بار تکرار. ($1 \le t \le Nt$)

الف) موقعیت جدید هر پراب به وسیله‌ی رابطه‌ی (۱۲) به دست می‌آید.

$$POP_p^t = POP_p^{t-1} + A_p^{t-1} \qquad (۱۲)$$

ب) بازگرداندن پراب ها به داخل فضای جستجو در صورت داشتن پراب انحراف یافته.

$$\begin{aligned} &if \ POP_p^t < X_{\min} \\ &then \ POP_p^t = \max\left\{X_{\min} + F_{rep}\left(POP_p^{t-1} - X_{\min}\right), X_{\min}\right\} \end{aligned} \qquad (۱۳)$$

$$\begin{aligned} &if \ POP_p^t > X_{\max} \\ &then \ POP_p^t = \min\left\{X_{\max} + F_{rep}\left(X_{\max} - POP_p^{t-1}\right), X_{\max}\right\} \end{aligned} \qquad (۱۴)$$

که $X_{\min}$ و $X_{\max}$ به ترتیب حداقل و حداکثر متغیرهای تصمیم می‌باشند.

(فاکتور تغییر موقعیت) (F_{rep}) که مقدار آن بین صـفر و یک است ($0 < F_{rep} < 1$)، به منظور بازگرداندن پراب‌هـای انحراف یافته به فضای تصمیم به کار می‌رود. F_{rep} در تکرار اول حلقه الگوریتم با یک مقدار اولیه (F_{rep}^{Init}) شروع می‌شود و در هر تکرار به F_{rep} به اندازه‌ی ΔF_{rep} اضافه مـی‌شـود. اگر F_{rep} آنقدر افزایش یافت که $F_{rep} \ge 1$ شد، آنگاه مقدار F_{rep} با یک مقدار حداقلی ($F_{rep}^{\min}$) جایگزین می‌شود.

$$\begin{aligned} &F_{rep} = F_{rep} + \Delta F_{rep} \\ &if \ F_{rep} \ge 1 + eps \quad then \quad F_{rep} = F_{rep}^{\min} \end{aligned} \qquad (۱۵)$$

ج) محاسبه برازندگی هر پراب.

$$M_p^t = f(POP_p^t) \qquad (۱۶)$$

د) محاسبه شتاب هر پراب برای تکرار بعدی.

$$A_p^t = G\sum_{\substack{k=1 \\ k\ne p}}^{N_p} u\left(M_k^t - M_p^t\right)\left(M_k^t - M_p^t\right)^{\alpha} \times \frac{\left(POP_k^t - POP_p^t\right)}{\left\|POP_k^t - POP_p^t\right\|^{\beta}} \qquad (۱۷)$$

ه) بروزرسانی محتویات مخزن:

این به‌روزرسانی ها شامل جادادن همه‌ی پراب‌های غیر چیره شده فعلی در مخزن است. به طور همزمان هر پراب چیره نشده در فرآیند از مخزن حذف می‌شود.

و) به روز کردن مختصات هر پراب در ابرمکعب‌ها.

ز) کنترل حجم مخزن:

از آنجایی که ظرفیت مخزن محدود است. هر زمان که ظرفیت مخزن به حد نصاب رسید، ابرمکعب هایی که بیشترین پراب را درون خود دارند شناسایی کرده و به طور تصادفی نقاط مازاد بر ظرفیت مخزن را حذف می‌کند و بعد از آن ابرمکعب‌های جدید در مناطق مرورشده در فضای پاسخ ایجاد می‌کند.

ح) پایان حلقه.

۵. نتایج آزمایشات و مقایسه با دیگر روش‌ها

برای بررسی توانایی الگوریتم MOCFO در بهینه سازی توابع چندهدفه، آن را به برخی توابع آزمون استاندارد اعمال مـی‌کنـیم. توابـع آزمـونی کـه در اینجـا معرفـی و اسـتفاده مـی‌شـوند، توابـع Fonseca [۷]، Kursawe [۸] و schaffer [۹] هستند.

بـرای بـرآورد کمّـی از عملکـرد الگـوریتم بهینـه‌سـازی چندهدفـه و مقایسـه الگـوریتم هـا بـا یکـدیگر، معیارهـای استانداری در نظر گرفته می‌شود که در اینجا از دو معیار GD و SP استفاده می‌شود.

معیار GD

مفهوم GD [۲] توسط Van Veldhuizen و Lamont معرفی شده است [۱۰] که به عنوان معیاری برای اندازه‌گیری میزان نزدیکی پاسخ‌های غیرچیره شده توسط الگوریتم به مجموعه‌ی بهینه پرتو (جبهه پرتو حقیقی) به کار می‌رود و به صورت رابطه‌ی (۱۸) تعریف می‌شود.

---

[۱] Hypercubes

[۲] Generational distance

$$GD = \frac{\sum_{i=1}^{n} d_i^2}{n} \qquad (18)$$

که n تعداد پاسخ های یافت شده‌ی غیرچیره شده است و d_i فاصله اقلیدسی بین هرکدام از این اعضاء و نزدیک‌ترین عضو از مجموعه‌ی بهینه پرتو می‌باشد که در فضای هدف اندازه گیری می‌شود. واضح است که اگر *GD=۰* باشد، نشان می‌دهد که همه‌ی اعضاء در مجموعه‌ی بهینه پرتو می‌باشد.

معیار SP

مفهوم *SP*[1] توسط Schott معرفی شده است [۱۱] که به عنوان معیاری برای اندازه‌گیری پراکندگی پاسخ‌های یافت شده‌ی غیرچیره شده در طول جبهه پرتو به کار می‌رود و به صورت رابطه‌ی (۱۹) تعریف می‌شود.

$$SP = \sqrt{\frac{\sum_{i=1}^{n} \left(\bar{d} - d_i\right)^2}{n-1}} \qquad (19)$$

که n تعداد پاسخ های یافت شده‌ی غیرچیره شده است و d_i از رابطه‌ی (۲۰) بدست می‌آید.

$$d_i = \min_j \left(\left| f_1^i(x) - f_1^j(x) \right| + \left| f_2^i(x) - f_2^j(x) \right| \right), \quad i,j = 1,2,\ldots,n \qquad (20)$$

و $\bar{d}$ میانگین همه‌ی d_i ها است. واضح است که اگر *SP=۰* باشد، نشان می‌دهد که همه‌ی اعضای جبهه پرتو با فاصله‌ی یکسان از هم قرار دارند.

توابع آزمون

در این قسمت روابط توابع آزمون مورد استفاده در روش پیشنهادی و مقایسه نتایج حاصل از الگوریتم *MOCFO* با الگوریتم‌های *MOPSO* و *NSGA-II* آورده شده است. لازم به ذکر است که درتمام آزمایشات انجام شده، گزارش نتایج به دست آمده از انجام ۱۰۰۰ بار تکرار حلقه الگوریتم و ۲۰ بار تکرار مستقل الگوریتم می‌باشد.

تابع آزمون ۱ (*SCH*):

[1] Spacing

این تابع توسط *Schaffer* مطرح شده است [۹] و به صورت زیر بیان می‌شود:

$$\min \quad f_1(x) = x^2$$
$$\min \quad f_2(x) = (x-2)^2$$

به طوری که $-10^3 < x < 10^3$ می‌باشد.

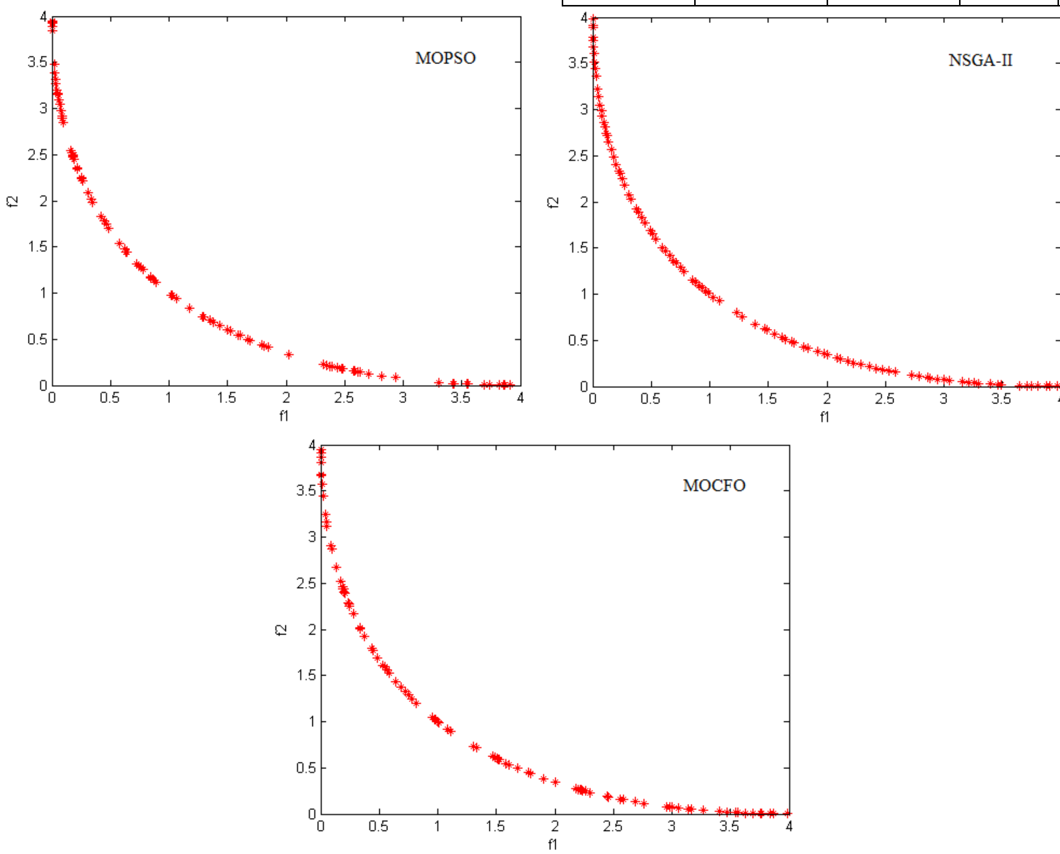
جبهه پرتوی تولید شده ی این تابع را توسط الگوریتم‌های MOCFO، MOPSO و NSGA-II در **شکل ۱** می‌توان مشاهده کرد.

در **جدول ۱** مقایسه‌ی بین نتایج این سه الگوریتم با توجه به معیارهای *GD* و *SP* که در بخش قبل توضیح داده شد، آورده شده است. این جدول نشان می‌دهد که مقادیر *GD* و *SP* به دست آمده با الگوریتم *MOCFO* خیلی نزدیک به مقادیر به دست آمده با الگوریتم‌های *MOPSO* و *NSGA-II* است وحتی از نظر معیار *SP* که میزان پراکندگی پاسخ‌ها را نشان می‌دهد، الگوریتم *MOCFO* از دو الگوریتم دیگر بهتر عمل کرده است.

جدول ۱- نتایج معیار *GD* و *SP* توسط *MOPSO*، *NSGA-II* و *MOCFO* برای اولین تابع آزمون

الگوریتم		*MOCFO*	*MOPSO*	*NSGA-II*
GD	حداقل	۰/۰۰۸۴۳۲	۰/۰۰۸۴۶۳	۰/۰۰۸۵۸۰
	حداکثر	۰/۰۰۹۶۱۸	۰/۰۰۹۴۹۱	۰/۰۰۹۴۴۵
	میانگین	۰/۰۰۸۹۸۶	۰/۰۰۸۹۶۹	۰/۰۰۸۹۵۷
	میانه	۰/۰۰۹۰۱۴	۰/۰۰۸۹۵۹	۰/۰۰۸۹۹۵
SP	حداقل	۰/۰۰۵۲۵۸	۰/۰۰۵۴۴۰	۰/۰۰۵۲۰۷
	حداکثر	۰/۰۰۶۱۲۴	۰/۰۰۶۵۰۳	۰/۰۰۶۳۴۳

	میانگین	۰/۰۰۵۷۵۴	۰/۰۰۵۸۷۸	۰/۰۰۵۷۹۵
	میانه	۰/۰۰۵۷۷۶	۰/۰۰۵۸۸۶	۰/۰۰۵۸۲۶



شکل ۱- جبهه پرتو تولید شده توسط MOPSO، NSGA-II و MOCFO برای اولین تابع آزمون

جبهه پرتوی تولید شده‌ی این تابع را توسط الگوریتم‌های MOCFO، MOPSO و NSGA-II در ***شکل ۲*** می‌توان مشاهده کرد.

در جدول ۲ مقایسه‌ی بین نتایج این سه الگوریتم با توجه به معیارهای GD و SP آورده شده است. این جدول نشان می‌دهد که مقادیر GD و SP به دست آمده با الگوریتم MOCFO نزدیک به مقادیر به دست آمده با الگوریتم‌های MOPSO و NSGA-II است.

تابع آزمون ۲ *(KUR)* :

این تابع توسط *Kursawe* مطرح شده است [۸] و به صورت زیر بیان می‌شود:

$$\min \quad f_1(\vec{x}) = \sum_{i=1}^{2}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$$

$$\min \quad f_2(\vec{x}) = \sum_{i=1}^{3}\left(|x_i|^{0.8} + 5\sin(x_i)^3\right)$$

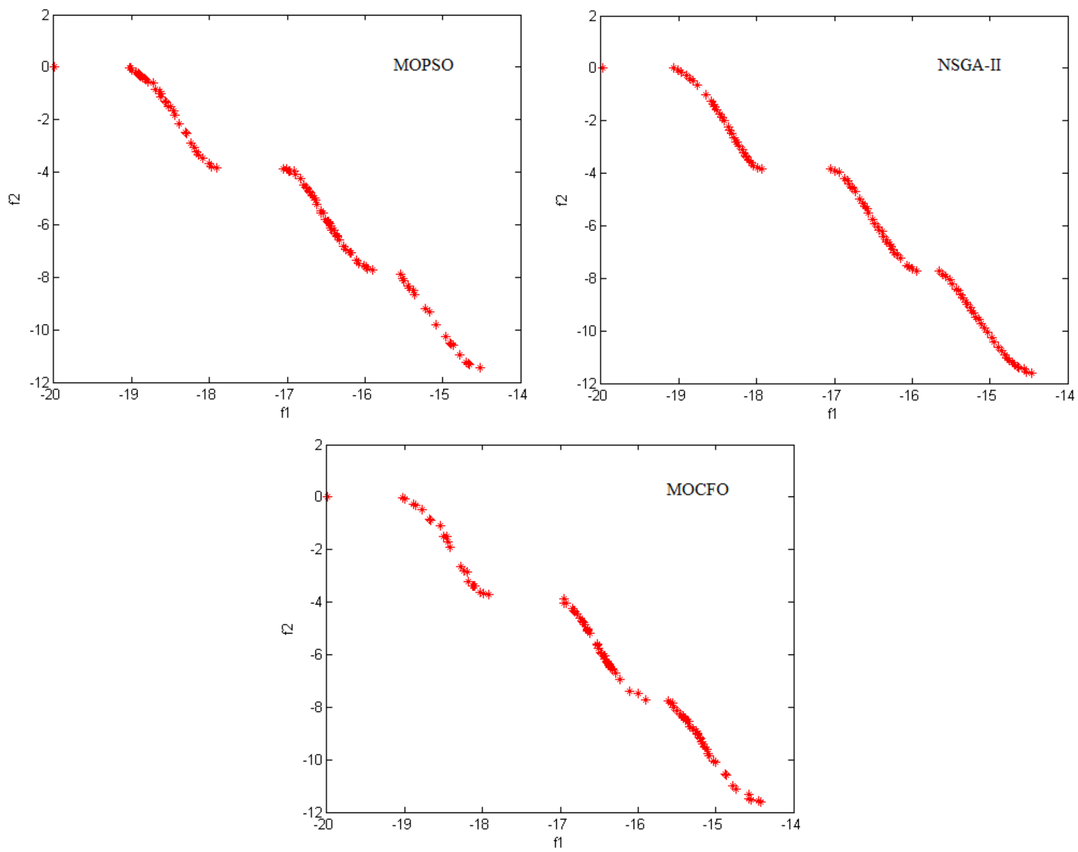
به طوری که $-5 < x_1, x_2, x_3 < 5$ می‌باشد.

جدول ۲- نتایج معیار GD و SP توسط MOPSO، NSGA-II و MOCFO برای دومین تابع آزمون

الگوریتم	*MOCFO*	*MOPSO*	*NSGA-II*

GD	حداقل	۰/۰۱۶۹	۰/۰۱۴۲	۰/۰۰۹۹
	حداکثر	۰/۰۲۰۷	۰/۰۱۷۳	۰/۰۱۲۱
	میانگین	۰/۰۱۹۰	۰/۰۱۵۹	۰/۰۱۰۸
	میانه	۰/۰۱۹۴	۰/۰۱۵۹	۰/۰۱۰۷

SP	حداقل	۰/۰۱۶۹	۰/۰۲۱۴	۰/۰۰۹۷
	حداکثر	۰/۰۶۸۴	۰/۰۳۰۶	۰/۰۱۵۶
	میانگین	۰/۰۵۴۶	۰/۰۲۶۲	۰/۰۱۱۸
	میانه	۰/۰۵۵۴	۰/۰۲۶۷	۰/۰۱۱۵



شکل ۲- جبهه پرتو تولید شده توسط MOPSO، NSGA-II و MOCFO برای دومین تابع آزمون

تابع آزمون۳ (*FON*):

این تابع توسط *Fonseca* و *Fleming* مطرح شده است [۷] و به صورت زیر بیان می‌شود:

$$\min \quad f_1(\vec{x}) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$$

$$\min \quad f_2(\vec{x}) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right)$$

به طوری که $-4 < x_1, x_2, x_3 < 4$ می‌باشد.

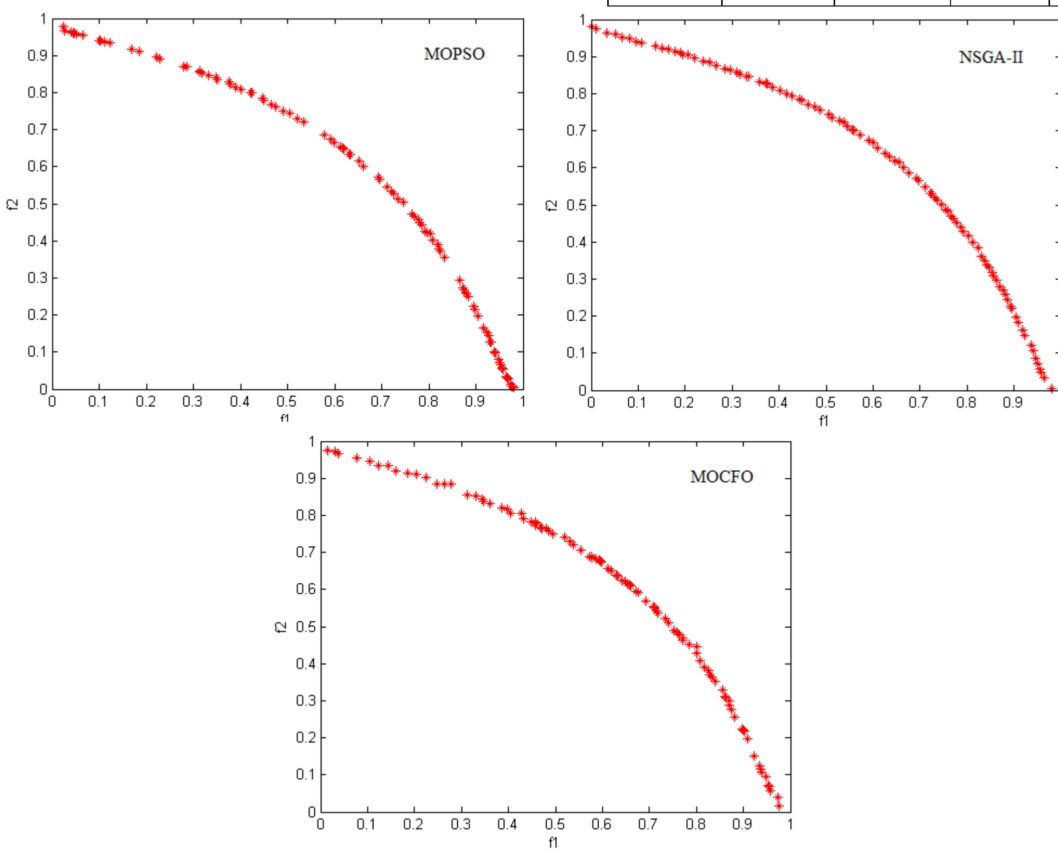
جبهه پرتوی تولید شده‌ی این تابع را توسط الگوریتم‌های MOCFO، MOPSO و NSGA-II در ***شکل ۳*** می‌توان مشاهده کرد.

در جدول ۳ مقایسه‌ی بین نتایج این سه الگوریتم با توجه به معیارهای GD و SP آورده شده است. این جدول نشان می‌دهد که مقادیر *GD* و *SP* به دست آمده با الگوریتم *MOCFO* نزدیک به مقادیر به دست آمده با الگوریتم‌های *MOPSO* و *NSGA-II* است.

جدول ۳- نتایج معیار GD و SP توسط MOPSO، NSGA-II و MOCFO برای سومین تابع آزمون

الگوریتم	*MOCFO*	*MOPSO*	*NSGA-II*

GD	حداقل	۰/۰۰۷۰	۰/۰۰۳۹	۰/۰۰۴۵
	حداکثر	۰/۰۰۷۷	۰/۰۰۵۲	۰/۰۰۵۲
	میانگین	۰/۰۰۷۴	۰/۰۰۴۴	۰/۰۰۴۹
	میانه	۰/۰۰۷۳	۰/۰۰۴۴	۰/۰۰۴۹
SP	حداقل	۰/۰۰۳۲	۰/۰۰۱۲	۰/۰۰۱۵
	حداکثر	۰/۰۰۵۳	۰/۰۰۲۱	۰/۰۰۲۵
	میانگین	۰/۰۰۳۸	۰/۰۰۱۵	۰/۰۰۲۱
	میانه	۰/۰۰۳۶	۰/۰۰۱۴	۰/۰۰۲۱



شکل ۳- جبهه پرتو تولید شده توسط MOPSO، NSGA-II و MOCFO برای سومین تابع آزمون

۶. نتیجه گیری

در این مقاله روش جدیدی در بهینه‌سازی چندهدفه مبتنی بر الگوریتم بهینه‌سازی نیروی مرکزی به نام الگوریتم بهینه‌سازی نیروی مرکزی چندهدفه (MOCFO) ارائه شده است. روش MOCFO از مفهوم "بهینگی پرتو" برای شناسایی موقعیت‌های "غیرچیره شده" و از یک "مخزن بیرونی" برای نگهداری این موقعیت‌ها استفاده می‌کند. برای اطمینان از صحت عملکرد روش ارائه شده در مواجه با مسائل بهینه‌سازی چندهدفه، برروی توابع استاندارد معتبر مورد آزمایش قرار گرفته است. نتایج نهایی حاصل از مقایسه روش MOCFO با دو الگوریتم پرکاربرد نشان داد که این روش توانایی نسبتا خوبی در بهینه‌سازی چندهدفه دارد و با دیگر روش های معروف قابل رقابت و مقایسه است.

مراجع

[۱] سید حمید ظهیری، "طبقه بندی کننده چند منظوره گروه ذرات،" نشریه مهندسی برق و مهندسی کامپیوتر ایران، سال۴، شماره۲، پاییز و زمستان ۱۳۸۵.

[۲] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, vol. ۶, no. ۲, pp. ۱۸۲–۱۹۷, ۲۰۰۲.

[۳] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” Proceedings of the IEEE

International Conference on Neural Networks, vol. ۴, pp. ۱۹۴۲–۱۹۴۸, Nov/Dec ۱۹۹۵.

[۴] R. A. Formato, “Central force optimization: A new deterministic gradient-like optimization metaheuristic,” OPSEARCH, Journal of the Operations Research Society of India, vol. ۴۶, no. ۱, pp. ۲۵–۵۱, ۲۰۰۹.

[۵] R. A. Formato, “Parameter-free deterministic global search with simplified central force optimization,” in: D. S. Huang, Z. Zhao, V. Bevilacqua, J. C. Figueroa, Eds., Advanced Intelligent Computing Theories and Applications, Springer-Verlag, Berlin, vol. ۶۲۱۵, pp. ۳۰۹–۳۱۸, ۲۰۱۰.

[۶] M. R. Sierra and C. A. C. Coello, “Multi-objective particle swarm optimizers: A survey of the state-of-the-art,” Int. J. Comput. Intell. Res., vol. ۲, no. ۳, pp. ۲۸۷–۳۰۸, ۲۰۰۶.

[۷] C. M. Fonseca and P. J. Fleming, “Multiobjective optimization and multiple constraint handling with evolutionary algorithms–Part II: An application example,” IEEE Transactions on Systems, Man, & Cybernetics Part A: Systems & Humans, vol. ۲۸, no. ۱, pp. ۳۸–۴۷, ۱۹۹۸.

[۸] F. Kursawe, “A variant of evolution strategies for vector optimization,” in Parallel Problem Solving from Nature, H.-P. Schwefel and R. Männer, Eds. Berlin, Germany: Springer-Verlag, vol. ۴۹۶, pp. ۱۹۳–۱۹۷, ۱۹۹۱.

[۹] J. D. Schaffer, “Multiple objective optimization with vector evaluated genetic algorithms,” in Proceedings of the First International Conference on Genetic Algorithms, J. J. Grefensttete, Ed. Hillsdale, NJ: Lawrence Erlbaum, pp. ۹۳–۱۰۰, ۱۹۸۷.

[۱۰] D. A. Van Veldhuizen and G. B. Lamont, “Multiobjective evolutionary algorithm research: A history and analysis,” Dept. Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol., Wright-Patterson AFB, OH, Tech. Rep. TR-۹۸-۰۳, ۱۹۹۸.

[۱۱] J. R. Schott, “Fault tolerant design using single and multicriteria genetic algorithm optimization,” M.S. thesis, Dept. Aeronautics and Astronautics, Massachusetts Inst. Technol., Cambridge, MA, May ۱۹۹۵.