

بهینه سازی گروه ذرات فازی مشارکتی

محمد رضا میبدی
دانشگاه صنعتی امیر کبیر
E-mail: mmeybodi@aut.ac.ir

محمد حسین نوروزی بیرامی
دانشگاه آزاد اسلامی واحد اسکو
E-mail: mh.noroozi@gmail.com

چکیده - بهینه سازی گروه ذرات یک تکنیک بهینه سازی مبتنی بر جمعیت می باشد که بر اساس قوانین احتمال کار می کند. در این روش هر یک از ذرات سعی می کنند به سمتی حرکت کنند که بهترین تجربه های فردی و گروهی در آن نقاط روی داده است. دو مشکل اصلی الگوریتم PSO استاندارد، افتادن در دام بهینگی محلی و پایین بودن سرعت همگرایی آن می باشد. یکی از روش های حل مشکلات عنوان شده استفاده از روش هایی ترکیبی مانند منطق فازی و یا الگوریتم های ژنتیک و... می باشد. ما در این مقاله یک الگوریتم ترکیبی با استفاده از منطق فازی ارائه می کنیم که از طرفی عیب اساسی PSO استاندارد را تا حد زیادی حل می کند و از طرف دیگر سرعت همگرایی الگوریتم را افزایش می دهد. الگوریتم جدید را برای استفاده در بهینه سازی گروه ذرات مشارکتی به کار می گیریم که اولاً امکان گسترده ای برای جستجوی موازی فراهم می کند و از طرف دیگر برای مقدار دهی ذرات از بهینه های محلی استفاده می کند که امکان همگرایی را به شدت افزایش می دهد. الگوریتم های محدودی برای بهینه سازی گروه ذرات مشارکتی ارائه شده است، برای اثبات برتری الگوریتم پیشنهادی در برابر الگوریتم های موجود از توابع محک استاندارد استفاده می کنیم. نتایج حاصل بیانگر برتری الگوریتم پیشنهادی نسبت به روش های موجود می باشد.

کلید واژه- بهینه سازی گروه ذرات، بهینه سازی گروه ذرات مشارکتی، بهینگی محلی، منطق فازی، FPSO.

۱- مقدمه

داند، یکی از بهترین استراتژی ها دنبال کردن پرنده ای می باشد که به غذا نزدیک تر است، این تئوری، استراتژی اصلی الگوریتم PSO می باشد.

در PSO هر پرنده یک جواب ممکن در فضای مسئله می باشد که ذره نامیده می شود. هر ذره دارای یک مقدار شایستگی می باشد که توسط تابع شایستگی مسئله محاسبه می شود. ذره ای که به جواب نزدیک تر باشد، شایستگی بیشتری دارد. این الگوریتم ماهیت پیوسته ای دارد و در کاربرد های متعددی کارایی خود را اثبات کرده است [1]. در این مقاله یک الگوریتم ترکیبی فازی با PSO به نام های FPSO ارائه شده است که اولاً مشکلات الگوریتم استاندارد را تا حد زیادی حل می کند و از طرفی باعث افزایش سرعت همگرایی آن می شود. برای اینکه بتوان امکان جستجوی موازی به الگوریتم PSO استاندارد اضافه شود، الگوریتم های بهینه سازی گروه ذرات مشارکتی ارائه شده است [18,24]. این الگوریتم ها با تقسیم فضای مسئله در بین دسته های ذرات مختلف، اولاً امکان جستجوی

در بسیاری از مسائل بهینه سازی بویژه مسائل بزرگ، انتخاب بهترین جواب از طریق جستجوی همه جانبه اگر چه غیر ممکن نیست، ولی کاری بسیار مشکل است. هدف مسائل بهینه سازی کاهش زمان این جستجوی می باشد. روش های ابتکاری راه حل های خوبی برای یافتن جواب بهینه می باشد، ولی تضمینی برای یافتن جواب بهینه نمی دهند. اما امروزه با بزرگ و پیچیده شدن مسائل، استقبال از روش های ابتکاری بطور چشم گیری افزایش یافته است [2].

بهینه سازی گروه ذرات^۱ یکی از تکنیک های بهینه سازی ابتکاری می باشد که بر مبنای جمعیت کار می کند. ایده اصلی این روش ابتدا در سال ۱۹۹۵ توسط دکتر کندی و دکتر ابرهارت [3] مطرح گردید که از رفتار دسته جمعی ماهی ها و پرندگان برای یافتن غذا الهام می گیرد. گروهی از پرندگان و ماهی ها در یک فضای تصادفی دنبال غذا می گردند و تنها یک تکه غذا وجود دارد و هیچ یک از پرندگان از محل غذا اطلاعی ندارد و فقط فاصله خود تا غذا را می

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (2)$$

در معادله (۱)، w ضریب اینرسی است که به صورت خطی کاهش می یابد که معمولاً در بازه $[0-1]$ می باشد. c_1 و c_2 ضرایب یادگیری یا شتاب می باشد که در بازه $[0-2]$ انتخاب می شود که در بیشتر موارد برای هر دوی آنها از مقدار 1.49 و یا 2 استفاده شده است [3,6,7]. دو عدد $rand_1$ و $rand_2$ نیز اعداد تصادفی در بازه $[0-1]$ می باشد. همچنین مقدار نهایی سرعت هر ذره برای جلوگیری از واگرایی الگوریتم به یک بازه محدود می شود. $v_{id} \in [-v_{max}, v_{max}]$. شرط خاتمه الگوریتم همگرایی تا حد معین و یا توقف بعد از تعداد معینی تکرار می باشد. معادله (۲) نیز بردار موقعیت فعلی ذره را با توجه به سرعت جدید آن بروز رسانی می کند.

سمت راست معادله (۱) از سه قسمت تشکیل شده است، قسمت اول ضریبی از سرعت فعلی ذره می باشد، قسمت دوم برای چرخش ذره به سمت بهترین تجربه شخصی و قسمت سوم نیز باعث چرخش ذره به طرف بهترین تجربه گروهی می باشد [2]. w برای برقراری موازنه بین جستجوی محلی و سراسری، برای اولین بار در [13] پیشنهاد گردید که ضریب حرکت در جستجوی سراسری را مشخص می کند. یکی از مشکلات شناخته شده الگوریتم PSO این است که برای ذره ای که شامل اطلاعات g_best است، قسمت های دوم و سوم معادله (۱)، صفر خواهد شد. بنابراین ذره در راستای بردار حرکت قبلی خود حرکت خواهد کرد. از طرفی چون به طور معمول w کوچکتر از یک می باشد، این قسمت نیز میرا خواهد بود. بنابراین ذره شامل g_best ثابت خواهد شد و سایر ذرات نیز به این ذره همگرا می شوند [2]. در بسیار از این موارد این موضوع باعث همگرایی زودرس الگوریتم به یک بهینه محلی خواهد شد. برای رفع این مشکل ابتدا در سال ۲۰۰۲، الگوریتم جدیدی با نام GCPSO^۲ ارائه گردید [8] که در این روش یک پارامتر جدیدی به الگوریتم اضافه شده بود که باعث جستجوهای تصادفی ذره g_best حول جواب بهینه خواهد شد. دومین مشکلی که الگوریتم PSO استاندارد دارد، همگرایی خطی آن است که ممکن است بسیار زمان بر باشد. ما در این مقاله دو روش ترکیبی برای حل این مشکلات ارائه می کنیم.

موازی را به PSO استاندارد اضافه می کنند و از طرف دیگر امکان همکاری بین دسته های ذرات مختلف را برای همگرایی بهتر به وجود می آورد. در این مقاله ما یک الگوریتم بهینه سازی گروه ذرات مشارکتی ارائه می کنیم که از یک طرف امکان جستجوی موازی را به وجود می آورد و از طرف دیگر برای مقدار دهی اولیه ذرات از بهینه های محلی فضای مسئله استفاده می کند.

بخش های بعدی این مقاله به این صورت سازماندهی شده است. در بخش دوم الگوریتم PSO استاندارد مورد بحث قرار گرفته است. در بخش سوم پیش زمینه ای از منطق فازی و کاربرد های آن ارائه می شود. در بخش چهارم الگوریتم های ترکیبی PSO با منطق فازی را توضیح خواهیم داد. در بخش پنجم الگوریتم های PSO مشارکتی توضیح داده شده و در بخش ششم الگوریتم پیشنهادی مورد بحث قرار خواهد گرفت. در بخش هفتم ارزیابی های انجام گرفته برای الگوریتم پیشنهادی با روش های موجود را بیان می کنیم و در نهایت در بخش هشتم نتیجه گیری را بیان خواهیم کرد.

۲- الگوریتم PSO استاندارد

PSO از جمله الگوریتم های جستجوی موازی مبتنی بر جمعیت است که با یک گروه از جواب های تصادفی (ذره ها) شروع به کار می کند، سپس برای یافتن جواب بهینه در فضای مسئله با به روز کردن مکان ذره ها به جستجو ادامه می دهد. هر ذره به صورت چند بعدی (بسته به نوع مسئله) با دو بردار V_{id} و X_{id} که به ترتیب معرف موقعیت مکانی و سرعت بعد d ام از i امین ذره هستند، مشخص می شود. در هر مرحله از حرکت جمعیت، مکان هر ذره با دو مقدار بهترین به روز می شود. اولین مقدار، بهترین تجربه ای است که خود ذره تا به حال بدست آورده است و با p_best نشان داده می شود. دومین مقدار، بهترین تجربه ای است که در بین تمامی ذره ها بدست آمده است و با g_best نشان داده می شود [4,5]. در هر تکرار، الگوریتم بعد از یافتن دو مقدار بالا، سرعت و موقعیت جدید ذره را بر اساس معادلات (۱) و (۲) بروز رسانی می کند.

$$v_{id}(t+1) = w \cdot v_{id}(t) + c_1 \cdot rand_1(p_best_{id}(t) - x_{id}(t)) + c_2 \cdot rand_2(g_best_{id}(t) - x_{id}(t)) \quad (1)$$

۳- پیش زمینه ای در مورد سیستم های فازی

منطق فازی تکنولوژی نسبتاً جدیدی است که در مقابل روشهای مرسوم برای طراحی و مدلسازی سیستمی که نیازمند ریاضیات و احتمالات پیشرفته و نسبتاً پیچیده می باشد، به کار می رود [9,10]. مرحله پردازش که به آن موتور استنباط نیز گفته می شود بر اساس مجموعه ای از قوانین فازی بصورت عبارات IF-THEN عمل می کند. دو رویه عمده برای استنباط وجود دارد. نوع اول مدل استنباط ممدانی و نوع دوم، مدل تاکاگی سوگنو می باشد. این دو روش در بسیاری از جنبه ها یکسان می باشند، مانند فازی سازی ورودیها و عملگرهای فازی، اما تفاوت اصلی بین روش سوگنو و ممدانی در شکل خروجی آنها می باشد [11]. ما در طراحی سیستم مورد نظر، بر حسب نتایج مورد انتظار از روش آقای ممدانی استفاده کرده ایم.

۴- روش ترکیبی فازی با PSO

با توجه به مطالب ارائه شده در بخش قبل و آشنایی با مشکلات PSO می توان گفت که از جمله مشکلات مهم روش استاندارد یکی افتادن در دام بهینگی محلی و دیگری پایین بودن سرعت همگرایی آن است. برای حل مشکل اول باید راهکاری اندیشید تا بهینگی محلی تشخیص داده شده و از آن اجتناب شود [12]. روش های مختلفی برای این کار پیشنهاد شده است. در [13] برای اجتناب از افتادن در دام بهینگی محلی در صورتی که سرعت ذره از یک حد مشخصی کمتر شود ولی میزان شایستگی جواب بدست آمده قابل قبول نباشد از یک تابعی برای ایجاد یک شک در سرعت ذره استفاده می شود. در [14,15] از یک تابع غیر خطی فازی برای تغییر ضریب اینرسی استفاده می شود، در صورتی که میزان شایستگی بدست آمده قابل قبول نباشد و ضریب اینرسی نیز کاهش یافته باشد، موجب افزایش این ضریب شده و با افزایش سرعت ذره امکان جستجوی سراسری را بیشتر می کند. در [16,17] با شناسایی دو قله محلی نزدیک به هم تا حد امکان از افتادن در دام بهینگی محلی حفظ می کند. همه این روش های ارائه شده به نوعی مانع از افتادن در دام بهینگی محلی خواهد شد.

۴-۱- روش ترکیبی FPSO

ابتدا یک الگوریتم ترکیبی به نام FPSO معرفی می کنیم که ضرایب شتاب c1 و c2 برای این الگوریتم یک تابع فازی

در نظر گرفته شده است. سه پارامتر d1، d2 و NCBPE^۴ به عنوان ورودی این تابع و c1 و c2 خروجی آن می باشد. با توجه با اینکه در اکثر منابع، دو ضریب c1 و c2 به صورت مساوی در نظر گرفته شده است، ما نیز در این الگوریتم این دو مقدار را مساوی هم در نظر می گیریم و تابع فازی پیشنهادی فقط یک خروجی دارد.

$$d1 = |p_best - x| \quad d2 = |g_best - x| \quad (۳)$$

$$NCBPE = \frac{CBPE_{min} - CBPE}{CBPE_{max} - CBPE_{min}} \quad (۴)$$

در روابط (۳) دو پارامتر d1 و d2 معرف میزان نزدیکی ذره به بهترین تجربه محلی و سراسری را نشان می دهد که می تواند میزان نزدیکی به بهینه محلی و سراسری را بازگو نماید و NCBPE نیز بهترین ارزیابی کارایی فعلی نرمال شده را مشخص می نماید [7,8] که در رابطه (۴) مشخص شده است. $CBPE_{min}$ بهترین جوابی است که تا کنون بدست آمده و در مقابل $CBPE_{max}$ بدترین جوابی است که تا کنون بدست آمده است. اما نکته مهمی که در این الگوریتم وجود دارد، انتخاب قوانین فازی می باشد که تاثیر مستقیم بر روی نتایج بدست آمده دارد. جدول ۱ تعدادی از قوانین استفاده شده در این سیستم را نشان می دهد.

جدول ۱: تعدادی از قوانین فازی الگوریتم FPSO

Rules	Input			Output
	d1	d2	NCBPE	c1 & c2
1	High	High	High	Low
2	Low	Low	Low	High
3	Low	Low	High	Low
4	Low	Low	Medium	Medium

با توجه به توضیحاتی که در بخش های قبلی ارائه شد، چون c1 و c2 ضرایب شتاب برای افزایش جستجوی محلی حول بهترین تجربه گروهی و فردی می باشد. بنابراین کاهش c1 و c2 باعث افزایش جستجوی سراسری و افزایش آن باعث افزایش جستجوی محلی می باشد. در نتیجه در این تابع، نمودار c1 و c2 نموداری صعودی خواهد بود تا رفته رفته از میزان جستجوی سراسری کم شده و جستجوی محلی افزایش پیدا کند. قوانین ارائه شده در جدول ۱ نیز بر این اصل استوار می باشد. قوانین ارائه شده به طور واضح مبنای تصمیم گیری را مشخص می کند ولی باید به این نکته

اصلی و p_g^S بهترین ذره فعلی در دسته های پیرو را نشان می دهد. g_{best}^M نیز بهترین مقدار شایستگی بدست آمده از ذرات دسته اصلی و g_{best}^S بهترین مقدار شایستگی بدست آمده از ذرات دسته های پیرو می باشند. برای اینکه بتوان تاثیر مقادیر شایستگی را کنترل کرد، می توان مقدار φ را از رابطه (۶) محاسبه کرد.

$$\varphi = \begin{cases} 0 & g_{best}^S < g_{best}^M \\ 0.5 & g_{best}^S = g_{best}^M \\ 1 & g_{best}^S > g_{best}^M \end{cases} \quad (6)$$

دسته دوم MCP SO همکاری کننده^۸ نامیده می شود [15] که در این روش دسته اصلی با همکاری دسته های پیرو موقعیت و سرعت ذرات خود را بروز رسانی می کند. رابطه (۷) برای بروز رسانی سرعت ذره در این نوع همکاری به کار می رود [19,20].

$$v_{id}^M = wv_{id}^M(t) + r_1c_1(p_{id}^M - x_{id}^M) + r_2c_2(p_{gd}^M - x_{id}^M) + r_3c_3(p_{gd}^S - x_{id}^M) \quad (7)$$

۵-۲- بهینه سازی گروه ذرات مشارکتی چند دسته ای

روش دیگری نیز برای CPSO ارائه گردیده [24]، که CPSO-S_k نامیده شده است. به طوری که می توان یک مفهوم خیلی ساده برای الگوریتم PSO اضافه کرد. در PSO استاندارد از یک بردار n بعدی استفاده می کنیم که می توان آن را به n بردار یک بعدی تقسیم بندی نماییم و هر بردار را به یک دسته واگذار نماییم. هر دسته باید یک بعد از جواب نهایی را پیدا نمایند. در واقع هر دسته مسئول بهینه سازی یک مسئله یک بعدی می باشد. یکی از پیچیدگی های این مسئله این است که برای کمینه کردن تابع شایستگی نیاز به یک بردار n بعدی داریم [21]. اگر هر یک از دسته ها یک بعد از جواب را برای محاسبه میزان شایستگی بردار جواب ارائه نمایند، واضح است که در این صورت استقلال دسته ها از بین خواهد رفت. بنابراین نیاز به یک بردار مفهومی داریم که مفهوم مناسبی را برای تک تک دسته ها برای محاسبه میزان شایستگی ارائه نماید [23]. ساده ترین روش این است که برای تشکیل این بردار n بعدی، بهترین تجربه هر یک از دسته ها را بکار برد. بنابراین

توجه کرد که قانون سوم، قانونی است که بهینه گی محلی را تشخیص می دهد به این صورت که ذره هم به بهترین جواب محلی و هم به بهترین جواب سراسری نزدیک شده است و از طرف دیگر جواب بدست آمده کارایی لازم را ندارد، در واقع یک بهینه محلی برای مسئله می باشد، بنابراین با کم کردن پارامتر خروجی باعث می شود که ذره از حالت کنونی خارج شده و بتواند به جستجوی سراسری ادامه دهد.

۵- الگوریتم های بهینه سازی گروه ذرات مشارکتی^۵

در این روش ها که تعمیمی بر روش های PSO استاندارد می باشد، به این صورت عمل می شود که به جایی اینکه از یک دسته ذره برای حل مسئله استفاده شود، چندین دسته ذره برای پیدا کردن جواب بهینه به صورت مشارکتی با هم کار می کنند، روش های مختلفی برای این کار پیشنهاد شده است که در ادامه دو نوع پرکاربرد این روش ها را ارائه می نماییم.

۵-۱- بهینه سازی گروه ذرات مشارکتی ارباب رعیتی

در این روش جمعیت از یک دسته اصلی و چند دسته پیرو تشکیل می شود و رابطه بین دسته های اصلی و پیرو می تواند امکان رسیدن به بهینه ترین جواب را برای دسته اصلی به وجود می آورد. به این روش مدل ارباب رعیتی^۶ گفته می شود [18]. هر یک از دسته های پیرو الگوریتم PSO را به صورت مجزا اجرا می کند و موقعیت و سرعت اعضای خود را بروز رسانی می کند. بعد از اینکه تک تک دسته های پیرو موقعیت جدید خود را مشخص کردند، بهترین نتایج بدست آمده محلی را برای دسته اصلی ارسال می کنند. دسته اصلی می تواند به دو صورت متفاوت با این نتایج رفتار نماید. روش اول روش، رقابتی^۷ می باشد که دسته اصلی برای بروز رسانی سرعت از رابطه (۵) استفاده می کند.

$$v_i^M(t+1) = wv_i^M(t) + r_1c_1(p_i^M - x_i^M(t)) + \varphi r_2c_2(p_g^M - x_i^M(t)) + (1-\varphi)r_3c_3(p_g^S - x_i^M(t)) \quad (8)$$

در روابط (۵) و (۶) M معرف دسته اصلی و S معرف دسته های پیرو می باشد و p_g^M بهترین ذره فعلی در دسته

هر دسته $n-1$ مقدار از سایر دسته ها دریافت کرده و برای $n-1$ دسته هم بهترین تجربه خود را ارسال نماید. الگوریتم CPSO-S برای اولین بار در [22] ارائه گردید.

۶- الگوریتم پیشنهادی

حال می خواهیم تعمیم جدیدی برای الگوریتم PSO و یا به بیان بهتر برای الگوریتم های FPSO ارائه نماییم که اولاً مشکلات افتادن در دام بهینگی محلی را به مراتب کاهش می دهد و ثانیاً پایین بودن سرعت همگرایی را بهبود ببخشد و همچنین مقدار دهی تصادفی اولیه را از بین می برد و از طرف دیگر به هدف اصلی روش های CPSO که امکان موازی سازی است نائل می شود. شکل ۱ شبه کدی برای الگوریتم CFPSO^۹ می باشد. الگوریتم ارائه شده شامل دو قسمت اصلی می باشد. قسمت اول شامل n دسته ذرات مستقل می باشد. در مرحله اول فضای جستجو را به n قسمت مستقل تقسیم می کنیم و هر دسته ذرات مستقل جستجو در یک قسمت می باشد. بعد از این تقسیم بندی هر دسته، الگوریتم FPSO را بر روی ناحیه مورد جستجو اعمال می کند و بعد از اینکه ۱۰٪ از کل جستجو ها را انجام دادند، کار دسته های اولیه تمام می شود و بهترین جوابی که برای هر دسته بدست آمده است به عنوان جواب های اولیه دسته اصلی محسوب می شود. بعد از اتمام این مرحله، دسته اصلی را با n ذره مقدار دهی می کنیم که مقدار اولیه آنها، بهترین جواب های بدست آمده از تک تک دسته های مرحله اول می باشد. به بیان دیگر بهترین تجربه گروهی تک تک دسته ها، بهینه های محلی هر ناحیه مستقل می باشد و دسته اصلی با استفاده از بهینه های محلی فضای جستجو مقدار دهی می شود. بعد از این کار ۹۰٪ تکرار های باقیمانده را با الگوریتم FPSO بر روی کل فضای جستجو و یا فقط بر روی ناحیه ای که بهترین جواب متعلق به آن است، اعمال می شود.

۷. ارزیابی الگوریتم های پیشنهادی

برای مقایسه الگوریتم پیشنهادی با الگوریتم های موجود مشارکتی که در بخش پنجم توضیح داده شد، شبیه سازی های زیادی با نرم افزار MATLAB 7.1 انجام گرفته و با استفاده از توابع محک استاندارد مورد ارزیابی قرار گرفته است این توابع، توابع پرکاربرد می باشد که در ارزیابی

های مشابه مورد استفاده قرار گرفته است و در جدول ۲ نشان داده شده است [2,7,9,10,11,15].

در این ارزیابی، ۲۰ ذره با ۱۰، ۲۰ و ۳۰ بعد و $c_1=c_2=2$ استفاده شده است. w برای توابع بخش ۵ در بازه [0.9 0.3] در نظر گرفته شده است و بعد از ۴۰۰۰ بار تکرار نتایج مورد نظر بدست آمده است. این شبیه سازی ها را ۱۰ بار به طور مستقل انجام داده و نتایج بدست آمده میانگین ده مرحله اجرای مستقل الگوریتم ها می باشد [18]. جداول ۳ نتایج این شبیه سازی ها را برای توابع محک مختلف و ابعاد مختلف نشان می دهد. این جدول بهترین جواب و میانگین جواب های بدست آمده برای الگوریتم های پیشنهادی و الگوریتم های موجود را نشان می دهد. شکل ۲ نمودار شایستگی تعدادی از توابع محک را برای ۳۰ بعد نشان می دهد. نتایج ارائه شده در جدول ۳ و شکل ۲ برتری الگوریتم پیشنهادی CFPSO را در برابر روش های موجود نشان می دهد. از مقایسه این نتایج می توان نتیجه گرفت که رفتار تابع پیشنهادی خیلی بهتر از دو الگوریتم COM-MCPSO و COL-MCPSO می باشد. این برتری هم در بهترین جواب و هم در متوسط جواب های بدست آمده دیده می شود. اما الگوریتم CPSO-Sk نتایج بهتری نسبت به دو الگوریتم قبلی ارائه کرده است. با توجه به اینکه الگوریتم CPSO-Sk یک مسئله n بعدی را به n مسئله یک بعدی تقسیم می کند، بنابراین همگرایی سریعتر از سایر روش ها می باشد و متوسط جواب های بدست آمده نیز بهتر از الگوریتم پیشنهادی می باشد. ولی با توجه به اینکه الگوریتم پیشنهادی فضای مسئله را به n زیر فضا تقسیم می کند و هر دسته، جستجو های ابتدایی را در این زیرفضا ها انجام می دهند، بنابراین بهینه های محلی مسئله به راحتی پیدا می شود. دسته اصلی از این بهینه های محلی برای مقدار دهی اولی ذرات استفاده می کند. در نتیجه جواب های الگوریتم CFPSO بهتر از روش های موجود، خصوصاً الگوریتم CPSO-Sk می باشد.

۸- نتیجه گیری

با توجه به نتایج بدست آمده، می توان گفت که استفاده از توابع فازی رفتار الگوریتم PSO استاندارد را به مراتب بهبود می بخشد و این امکان را برای الگوریتم PSO فراهم می کند که در مقابل موقعیت های غیر عادی، عکس العمل مناسبی نشان دهد. همچنین الگوریتم های PSO مشارکتی

بهینه های محلی، مشکل اصلی الگوریتم های PSO که افتادن در دام بهینگی محلی می باشد به مراتب مرتفع کردیم.

جدول ۲: توابع محک استاندارد

نام تابع	فرمول	نام تابع	فرمول
Ackley	$f(x) = 20 + e - 20e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^n x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^n \cos(2\pi x_i)}$	Schwefel's Problem 2.22	$f(x) = \sum_{i=1}^n x_i + \prod_{i=1}^n x_i $
Quadric	$f(x) = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	Rosenbrock	$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$
Rasterigin	$f(x) = \sum_{i=1}^n (x_i^2 - 10\cos(2\pi x_i) + 10)$	Quadric i.e. Noise	$f(x) = \sum_{i=1}^n i x_i^4 + \text{random}[0,1]$
Sphere	$f(x) = \sum_{i=1}^n x_i^2$	Generalized Schwefel's Problem 2.26	$f(x) = \sum_{i=1}^n x_i \sin(x_i)$

CFPSO Algorithm

define

gbest: array [1..n] k-dimension vector // for all swarms

Split search space to n independent area

// n is numbers of swarms and each swarm to one area P_j $j \in [1..n]$ initialize n k-dimensions PSOs:

for all swarms

for i=1:t // t is 1/10 of all iteration in simulation

apply FPSO // each swarm are limited in own area

update gbest array for each swarm

end

end

initialize k-dimension PSO with gbest values // swarm have n particle

select the best area with minimum fitness for next search

for i=t+1:end of simulation

apply FPSO for search area or selected area

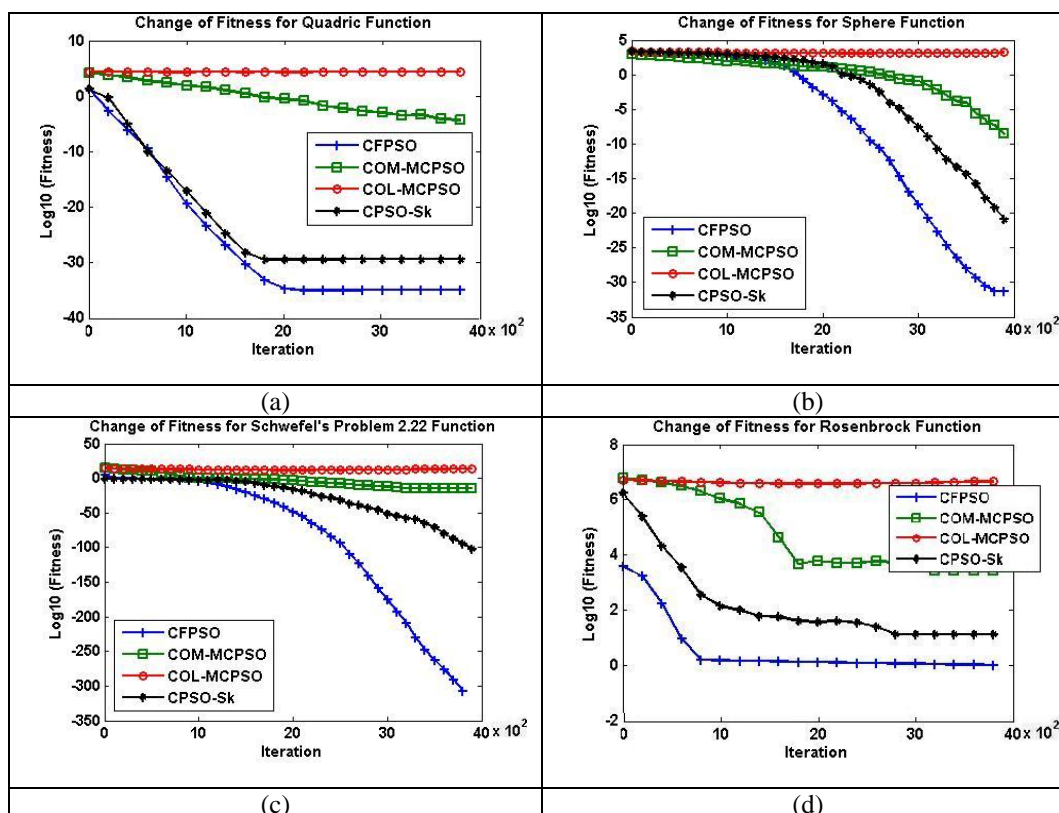
end

شکل ۱: شبه کد الگوریتم CFPSO

جدول ۳: نتایج بدست آمده از مقایسه روش های مشارکتی موجود با الگوریتم پیشنهادی با ابعاد مختلف

توابع محک	الگوریتم	۱۰ بعد		۲۰ بعد		۳۰ بعد	
		مقدار متوسط	بهترین مقدار	مقدار متوسط	بهترین مقدار	مقدار متوسط	بهترین مقدار
Ackley	CFPSO	0.1036	3.54e-16	0.0690	8.88 e-16	0.0519	1.14 e-14
	COM-MCP SO	0.4702	2.71 e-15	0.3128	6.21 e-15	0.2217	2.64 e-14
	COL-MCP SO	10.9454	8.09	16.1984	15.56	18.4521	17.49
	CPSO-S _k	0.4463	7.54 e-16	0.1774	2.66 e-15	0.1028	4.52 e-13
Quadric	CFPSO	5.48	9.98 e-162	7.53	1.21 e-71	9.55	7.74 e-36
	COM-MCP SO	431.21	3.42 e-46	1224.45	7.27 e-8	2187.81	3.23 e-5
	COL-MCP SO	7697.54	96.89	10758.34	803.91	24673.21	20200
	CPSO-S _k	5.11	5.80 e-142	7.00	1.40 e-58	9.49	5.35 e-29
Rasterigin	CFPSO	1.5688	0.0315	1.7901	0.0587	2.0770	0.0891
	COM-MCP SO	5.3896	0.2171	6.6436	0.2333	7.5615	0.8903
	COL-MCP SO	14.0205	9.8521	14.7710	10.8521	15.0863	11.7829
	CPSO-S _k	0.2051	0.0514	0.2235	0.0785	0.2608	0.0963
Sphere	CFPSO	104.80	7.31 e-96	167.43	4.99 e-46	241.47	6.10 e-34
	COM-MCP SO	98.98	2.35 e-45	118.23	8.52 e-37	136.31	9.71 e-12
	COL-MCP SO	1346.71	4.42	1658.54	242.05	1834.42	834.73
	CPSO-S _k	28.43	8.21 e-83	83.49	2.83 e-40	143.48	3.01 e-23
Schwefel's Problem 2.22	CFPSO	1.5 e+3	0	1.59 e+5	0	4.69 e+8	0
	COM-MCP SO	3.11 e+4	1.48 e-75	1.33 e+8	3.18 e-19	1.36 e+12	1.02 e-13

6.65 e+15	48.15	2.34 e+9	16.53	7.28 e+4	6.04	COL-MCPSO	Rosenbrock
5.27 e+13	0	1.80 e+7	0	3.53 e+3	0	CPSO-S _k	
5.63 e+4	19.64	9.03 e+3	5.50	1.34 e+3	2.03 e-6	CFPSO	
1.02 e+7	80.32	3.00 e+6	30.25	7.10 e+5	0.3697	COM-MCPSO	
2.81 e+10	1.36 e+7	4.58 e+7	2.40 e+6	3.04 e+6	4.21 e+2	COL-MCPSO	
1.32 e+5	24.28	3.05 e+4	9.00	9.84 e+3	6.11 e-3	CPSO-S _k	Quadric i.e. Noise
0.1262	1.40 e-4	0.1093	1.18 e-4	0.0831	1.04 e-4	CFPSO	
0.9002	1.05 e-2	0.4991	5.30 e-3	0.2360	9.86 e-4	COM-MCPSO	
10.6254	7.45	3.5810	1.72	0.6995	0.1319	COL-MCPSO	
0.1464	6.92 e-3	0.9814	2.74 e-3	0.0826	4.70 e-4	CPSO-S _k	
-239.06	-14888	-314.77	-9936	-395.11	-4970	CFPSO	Generalized Schwefel's Problem 2.26
-91.46	-7274	-174.14	-7063	-210.04	-4168	COM-MCPSO	
-66.74	-6002	-96.10	-5617	-111.74	-3382	COL-MCPSO	
-280.16	-12499	-494.55	-9299	-534.37	-4863	CPSO-S _k	



شکل (۲) نمودار میزان شاسستگی الگوریتم های مختلف برای تعدادی از توابع محک استاندارد. (a) تابع Quadric (b) تابع Sphere (c) تابع Rosenbrock (d) Schwefel's Problem 2.22

- [3] J. Kennedy and R. Eberhart, "Particle swarm optimization", Proceedings of IEEE International Conference on Neural Networks, vol. 4, 1995.
- [4] X. Hu and Y. Shi and R. Eberhart, "Recent advances in particle swarm optimization", IEEE, 2004.
- [5] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in Proc. IEEE Congr. Evol. Comput., Honolulu, HI, 2002, pp. 1671–1676.

مراجع

- [۱] حسین نظام آبادی پور و مجید رستمی شهر بابکی، "تعمیمی بر الگوریتم GCBPSO"، دوازدهمین کنفرانس مهندسی کامپیوتر ایران، اسفند ۱۳۸۵، صفحات ۲۹-۳۵.
- [۲] محمد شبانی و محمد رضا میبیدی، "PSO-LA: یک مدل جدید برای بهینه سازی"، دوازدهمین کنفرانس مهندسی کامپیوتر ایران، اسفند ۱۳۸۵، صفحات ۱۱۶۲-۱۱۶۹.

- Systems, Hang Zhou, China, 2007, pp. 15-17.
- [20] M. Song and G. Gu and X. Wang and R. Zhang, "Function approximation using the cooperative pso neural network", Harbin Engineering University, 2005.
- [21] F. V. D. Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization", IEEE Transactions On Evolutionary Computation, Vol. 8, No. 3, 2004.
- [22] F. V. D. Bergh and A. Engelbrecht, "Cooperative learning in neural networks using particle swarm optimizers", South African Comput., vol. 26, 2000, pp. 84-90.
- [23] M. Friedman and L. S. Savage, "Planning experiments seeking minima", in Selected Techniques of Statistical Analysis for Scientific and Industrial Research, and Production and Management Engineering, C. Eisenhart, M. W. Hastay, and W. A. Wallis, Eds. New York: McGraw-Hill, 1947, pp. 363-372.
- [24] F. V. D. Bergh and A. P. Engelbrecht. "Effects of swarm size on cooperative particle swarm optimizers". In Proceedings of the Genetic and Evolutionary Computation Conference. San Francisco. USA, 2001.
- [25] M. Clerc and J. Kennedy, "The particle swarm: explosion, stability, and convergence in a multidimensional complex space", IEEE Trans. Evolut. Comput. 6. 2002, pp. 58-73.
- [6] X. Cui, "Document clustering using particle swarm optimization", IEEE, 2005, pp. 185-191.
- [7] L. Hongbo and M. Abraham, "Fuzzy adaptive turbulent particle swarm optimization", IEEE, 2005, pp. 39-47.
- [8] F. Bergh and A. Engelbrecht, "A new locally convergent particle swarm optimizer", IEEE, 2002.
- [9] J. Liebowitz, The Handbook Of Applied Expert Systems. CRC Press LLC, ISBN: 0849331064, 1999.
- [10] C. Krishnamoorthy and S. Rajeev, Artificial Intelligence and Expert Systems for Engineers, CRC Press LLC, ISBN: 0849391253, 1996.
- [11] T. Kiink and J. S. Vesterstroem and J. Riget, "Particle swarm optimization with spatial particle extension", Proceedings of the IEEE Congress on Evolutionary Computation, 2002, pp. 1474-1479.
- [12] L. Hongbo and M. Abraham, "Fuzzy adaptive turbulent particle swarm optimization", IEEE, 2005, pp. 39-47.
- [13] Y. Shi and R. Eberhart, "A modified particle swarm optimizer", IEEE International Conference on Evolutionary Computation, USA, 1998, pp. 255- 262.
- [14] R. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms", Proceedings of IEEE Congress on Evolutionary Computation, Seoul, Korea, 2001, pp. 94-97.
- [15] B. Niu and Y. Zhu and X. Xian and H. Shen, "A multi-swarm optimizer based fuzzy modeling approach for dynamic systems processing", Elsevier, 2007.
- [16] M. Clerc and J. Kennedy, "The particle swarm: explosion, stability, and convergence in a multidimensional complex space", IEEE Trans. Evolut. Comput. 6. 2002, pp. 58-73.
- [17] R. Eberhart and S. Yuhui, "Fuzzy adaptive particle swarm optimization", IEEE, 2001, pp. 101-106.
- [18] B. Niu and Y. Zhu, X. He, H. Wu, "MCPSO: A multi-swarm cooperative particle swarm optimizer", Applied Mathematics and Computation 185, Elsevier, 2007, pp. 1050-1062.
- [19] Y. Xiong and J. Su, "Parallel cooperative particle swarm optimization based multistage transmission network planning", WSEAS International Conference on Instrumentation, Measurement, Circuits &

¹ - Particle Swarm Optimization - PSO

² - Guaranteed Convergence PSO - GCPSO

³ - Fuzzy PSO

⁴ - Normal Current Best Performance Evaluation - NCBPE

⁵ - Cooperative Particle Swarm Optimization

⁶ - master-slave model

⁷ - Competitive Multi-Swarm Cooperative PSO –
COM-MCPSO

⁸ - Collaborative Multi-Swarm Cooperative PSO –
COL-MCPSO

⁹ - Cooperative Fuzzy Particle Swarm Optimization -
CFPSO