

تطبیق پارامترهای گروه ذرات بهینه‌ساز مشارکتی با استفاده از

منطق فازی

زهرا افصحی^۱ محمد رضا میبیدی^۲

^۱ دانشکده مهندسی برق، کامپیوتر و فناوری اطلاعات دانشگاه آزاد اسلامی، قزوین، ایران

^۲ دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران

afsahi_ai@yahoo.com mmeybodi@aut.ac.ir,

چکیده: گروه ذرات بهینه‌ساز یک تکنیک بهینه‌سازی است و از حرکت و زندگی گروهی جانورانی مانند ماهی‌ها و پرندگان الهام می‌گیرد. از مشکلات عمده این الگوریتم قرار گرفتن در بهینه‌های محلی و بزرگی ابعاد است. در این مقاله مدل جدیدی با نام FCPSO-H، که ترکیبی از منطق فازی و الگوریتم گروه ذرات بهینه‌ساز CPSO-H میباشد پیشنهاد میگردد. در این مدل پارامترهای موثر در بروز رسانی سرعت و مکان هر ذره با استفاده منطق فازی تطبیق داده میشود. به منظور ارزیابی، الگوریتم پیشنهاد شده روی توابع نمونه استاندارد آزمایش شده است. نتایج به دست آمده نشان از برتری این الگوریتم در مقایسه با CPSO-H و PSO استاندارد دارد. آزمایشها نشان داده است که الگوریتم پیشنهادی با احتمال کمتری در بهینه‌های محلی گرفتار میشود.

Abstract: PSO is a population-based technique for optimization, which simulates the social behavior of the fish schooling or the bird flocking. Two significant weakness of this method are falling into local optimum and the curse of dimensionality. In this work, we present the FCPSO-H to overcome these weaknesses. Our approach was implemented in the cooperative PSO, which employs fuzzy logic to control the acceleration coefficients in velocity and position equation of each particle. The proposed approach is validated by function optimization problem from the standard literature. Simulation result indicates that, this approach is highly competitive with CPSO-H and PSO, specifically in its better general convergence performance.

کلمات کلیدی: گروه ذرات بهینه‌ساز، یادگیری مشارکتی، منطق فازی، بهینه‌سازی

۱- مقدمه

الگوریتم گروه ذرات بهینه‌ساز^۱ یک روش مکاشفه‌ای مبتنی بر جمعیت است که اولین بار توسط آقای کندی و ابرهات و در سال ۱۹۹۵ معرفی شد [1]. سرعت همگرایی و سادگی قوانین در تعیین جهت بهینه از ویژگی‌های مهم این الگوریتم به شمار می‌آیند. ایده اصلی در الگوریتم گروه ذرات بهینه‌ساز رفتار اجتماعی حیواناتی است که به صورت گروهی زندگی می‌کنند، مانند ماهی‌ها، زنبورها و پرندگان [1]. در این مدل هر عضو از جمعیت ذره نام دارد که در فضای جستجوی چند بعدی قرار می‌گیرد و در جهت‌ها و سرعت‌های مختلف برای رسیدن به یک موقعیت بهینه حرکت می‌کند. هر ذره برای حرکت در فضای جستجو، علاوه بر تجربه شخصی خود (بهترین موقعیتی که ذره در طول حیات خود ملاقات کرده) از بهترین تجربه ذرات گروه نیز استفاده می‌نماید. در واقع هر ذره یک راه حل مسأله بهینه‌سازی را نمایش می‌دهد. علت محبوبیت الگوریتم گروه بهینه‌ساز ذرات، سادگی به کارگیری و توانایی در همگرایی سریع به راه حل درست است. این دو عامل باعث شده که در زمینه‌های مختلف از جمله تهیه نقشه ترافیکی، طراحی مهندسی، بهینه‌سازی و سیستم‌های کامپیوتری و غیره مورد استفاده قرار بگیرد.

از سال ۱۹۹۵ تا کنون، الگوریتم‌های زیادی در زمینه توسعه دادن نسخه اصلی PSO در محیط‌های پیوسته و گسسته [14] [14] و حل مشکلات آن ارائه شده است که می‌توان به الگوریتم‌هایی جهت تنظیم پارامترها [13] [13, 17] [17] در معادلات سرعت، الگوریتم‌های مبتنی بر اجتماع که ساختارهای متفاوتی از توپولوژیهای اجتماعی برای همسایگی ذرات مطرح میکنند [2]، الگوریتم‌های ترکیبی که از ترکیب

^۱ Particle Swarm Optimization(PSO)

گروه ذرات بهینه‌ساز با سایر الگوریتم‌های بهینه‌ساز هوشمند از جمله الگوریتم‌های تکاملی [4]، اتوماتای یادگیر [12] [12] و منطق فازی به وجود آمده‌اند و استفاده از الگوریتم‌های چند جمعیتی که شامل مجموعه‌ای از گروه‌ها که با یکدیگر در تعامل هستند **Error! Reference source not found.** [3]، اشاره کرد. از جمله مشکلات الگوریتم PSO می‌توان قرار گرفتن در بهینه‌های محلی، بزرگی ابعاد، همگرایی زودرس [4] و از بین رفتن تنوع در بین جمعیت و ... را نام برد.

استفاده از منطق فازی [5] در الگوریتم PSO علاوه بر اینکه مشکل اصلی این الگوریتم که قرار گرفتن در بهینه‌های محلی است را بهبود می‌دهد، می‌تواند از دو جنبه مفید باشد. اولاً می‌توان از دانش موجود و تجربیات مثبت ذرات در تعیین روند تغییرات وزن میانی استفاده کرد، ثانیاً این روند با گرفتن بازخورد از اجرای الگوریتم اصلاح می‌شود. مزیت دوم باعث ایجاد پویایی و خودتنظیمی در این الگوریتم می‌شود. در [5] گونه‌ای جدید از PSO مبتنی بر منطق فازی ارائه شده است. در این مدل پارامتری با عنوان ضریب جذب تعریف شده است. این ضریب با توجه به تعداد ذرات بهینه در هر همسایگی و به صورت دستی قابل تنظیم است. در [7] [7] مدل دیگری از ساختار فازی در الگوریتم PSO پیشنهاد شده است. در این مدل بهینه‌ترین ارزش تابع شایستگی بدست آمده در طی تعداد مشخصی از تکرار، فاصله بین بهترین ذره عمومی و بهترین تجربه هر ذره و تعداد نسل‌هایی که در آن بهینه‌ترین شایستگی بدون تغییر است، به عنوان ورودی‌های سیستم فازی و ضرایب c_1 و c_2 و W سه خروجی این سیستم هستند. در [6] مدل دیگری از $FPSO^2$ ارائه شده است که در آن همسایگی ذرات به صورت فازی تعیین می‌شود. در این مدل نیز پارامتری تحت عنوان درجه عضویت هر ذره با توجه به مکانی که در آن قرار دارد، اضافه شده است. در [8] مدل جدیدی به نام $FATPSO^3$ ارائه شده است. در این مدل دو پارامتر آستانه به منظور کنترل مکان و سرعت ذرات در فضای جستجو تعریف شده است که خروجی‌های سیستم فازی می‌باشند. بهترین شایستگی بدست آمده و بردار سرعت نظیر به آن، ورودی‌های سیستم فازی در این مدل می‌باشد. در **Error! Reference source not found.** [15] الگوریتمی تحت عنوان $CFPSO^4$ معرفی شده است. ورودی‌ها در این الگوریتم، فاصله بهترین مکان پیدا شده توسط هر گروه تا مکان ذره جاری، فاصله بهترین تجربه هر ذره در گروه تا مکان ذره جاری و بدترین جواب به دست آمده و خروجی‌ها ضرایب c_1 و c_2 می‌باشند. در این الگوریتم تعداد قوانین فازی مورد استفاده ۴ عدد می‌باشد. در [16] نیز الگوریتمی مشابه ارائه شده است. ورودی‌ها و خروجی‌های سیستم فازی همانند آنچه که در [15] تعریف شده می‌باشند با این تفاوت که الگوریتم پایه در این مدل، PSO استاندارد است. در همگی این روشها پارامترهای جدیدی (ضریب جذب، تعداد نسل‌هایی که در آن شایستگی بدون تغییر است، درجه عضویت، پارامترهای آستانه و بدترین جواب بدست آمده در گروه‌ها) در الگوریتم PSO استاندارد تعریف شده است که سادگی الگوریتم PSO استاندارد تا حدی از بین برده است. از طرفی با اضافه شدن تعداد پارامترها بار محاسباتی الگوریتم نیز بالاتر می‌رود.

برای غلبه بر مشکل بزرگی ابعاد نیز الگوریتم‌هایی از جمله الگوریتم‌های $CPSO^5$ و $MCPSO^6$ گزارش شده‌اند. [18] در الگوریتم‌های $CPSO$ و $MCPSO$ تبادل اطلاعات بین ذرات از نوع همکاری و یا رقابتی است و یادگیری به شیوه مشارکتی است [9,10]. در الگوریتم $CPSO$ [10] ذرات در زیرگروه‌هایی قرار می‌گیرند و الگوریتم PSO استاندارد در هر زیرگروه به طور جداگانه اجرا می‌شود. البته لازم به ذکر است که در به روز رسانی ذرات در یک گروه، از اطلاعات بهینه گروه‌های دیگر نیز استفاده می‌شود. پس تبادل اطلاعات از نوع همکاری است. در مدل $MCPSO$ تبادل اطلاعات از نوع رقابتی است. در این مدل یک گروه اصلی وجود دارد و سایر گروه‌ها به صورت فرعی و در کنار این گروه اصلی در فضای جستجو حرکت می‌کنند.

در این مقاله، یک روش بهینه‌سازی با یک ساختار کنترل کننده فازی روی الگوریتم $CPSO-H$ پیشنهاد شده است. با استفاده از الگوریتم $CPSO$ و تقسیم کردن فضای جستجو به زیر فضاهایی با ابعاد کوچکتر به نوعی از تجمع ذرات در یک ناحیه ممانعت کرده و تنوع بین ذرات حفظ می‌شود. در واقع در این مدل سعی شده علاوه بر حل مشکل بزرگی ابعاد، بر مشکل اصلی الگوریتم PSO که قرار گرفتن در بهینه‌های محلی است حتی در یک بعد خاص غلبه شود [11]. نتایجی که بر روی ۸ تابع استاندارد آزمایش شده است نشان می‌دهد که این روش راندمان بالاتری نسبت به الگوریتم‌های PSO استاندارد و $CPSO-H$ دارد. در ادامه این مقاله و در بخش ۲ به معرفی الگوریتم PSO می‌پردازد. در بخش ۳ به بررسی الگوریتم $CPSO-H$ می‌پردازد. در بخش ۴ مدل $Fuzzy CPSO-H$ یا $FCPSO-H$ ارائه می‌شود. بخش پنجم اختصاص به ارائه نتایج شبیه سازی‌ها دارد و بخش پایانی نتیجه گیری می‌باشد.

²Fuzzy PSO

³Fuzzy Adaptive PSO

⁴Cooperative Fuzzy PSO

⁵Cooperative PSO

⁶Multi-Swarm Cooperative PSO

۲- الگوریتم گروه ذرات بهینه‌ساز استاندارد

بهینه سازی گروه ذرات الگوریتمی است که از رفتار اجتماعی موجوداتی که به صورت انبوه زندگی می کنند، الهام گرفته است. جمعیت اولیه این الگوریتم در یک بازه مشخص و به صورت تصادفی مقداردهی می‌شود. هر ذره در فضای D بعدی به عنوان یک راه حل از مساله محسوب می شود. ذره نام از گروه موقعیت X_i^d و سرعت V_i^d را در بعد d ام از فضای جستجو دارد. معادله به روز نمودن سرعت و موقعیت ذره در (۱) و (۲) نمایش داده شده است که جستجو بر اساس این فرمول‌ها برای رسیدن به یک بهینه شروع می‌شود. معادلات به روز رسانی مکان و سرعت برای i th ذره به شکل زیر به نمایش در می‌آید:

$$V_i^d(t+1) = V_i^d(t) + c_1 \text{rand}_1^d(t)(pbest_i^d(t) - X_i^d(t)) + c_2 \text{rand}_2^d(t)(gbest(t) - X_i^d(t)) \quad (1)$$

$$X_i^d(t+1) = X_i^d(t) + V_i^d(t+1) \quad (2)$$

که $pbest_i$ بهترین موقعیت ملاقات شده توسط ذره نام و $gbest$ بهترین موقعیت ملاقات شده توسط کل گروه است. برای تساوی (۱) سرعت تحت تاثیر دو عامل است. (۱) مکان $pbest$ هر ذره که به عنوان مشخصه درک و شناخت است (اکتشاف محلی). (۲) $gbest$ که مشخصه گروهی بودن است. (اکتشاف عمومی) و توسط V_{max} محدود شده است. rand_1^d و rand_2^d دو عدد تصادفی در بازه $[0,1]$ می باشند. C_1 و C_2 ضرایب مولفه های شناختی و اجتماعی می باشند.

ایراد الگوریتم گروه ذرات بهینه ساز استاندارد، نبود یک مکانیزم برای کنترل اندازه سرعت ذرات است. نبود چنین مکانیزمی خطر انشعاب و انفجار (خارج از محدوده) را افزایش می‌دهد. به منظور حل شدن این مسئله مشخصه‌ای به نام w معرفی شده است که تساوی (۱) را به شکل زیر

$$V_i^d(t+1) = wV_i^d(t) + c_1 \text{rand}_1^d(t)(pbest_i^d(t) - X_i^d(t)) + c_2 \text{rand}_2^d(t)(gbest(t) - X_i^d(t)) \quad (3)$$

تغییر می‌دهد: این مشخصه اغلب به طور خطی از ۰.۹ تا ۰.۴ در طول اجرا تغییر می‌کند. یکی از تغییرات مهمی که در الگوریتم گروه ذرات بهینه‌ساز استاندارد ایجاد شده، اضافه شدن فاکتور محدود کننده K است که توسط آقای کلرک و کندی ارائه شده است [11]. این پارامتر تابعی از دو مشخصه C_1 و C_2 است.

۳- الگوریتم گروه ذرات بهینه‌ساز مشارکتی

الگوریتم گروه ذرات بهینه‌ساز مشارکتی اولین بار توسط آقای ون-دن-برگ و انگلبرچ معرفی شد **Error! Reference source not found.** [9][9]. در این الگوریتم برای حل یک مسئله D بعدی، به جای تولید یک گروه D بعدی از S ذره، D گروه تک بعدی تشکیل می‌شود و هر گروه شامل S ذره است. در واقع هر گروه مسئول بهینه‌سازی یک مسئله یک بعدی است. در پایان هر گروه باید یک بعد از جواب نهایی را پیدا نماید. اگر هر گروه فقط یک بعد از فضای جستجو را بدون برقراری تعامل با سایر گروه‌ها ارائه کند، یا عبارتی انتخاب کردن بهترین راه حل از هر گروه در حالت تنها و مستقل از سایر گروه‌ها باشد، مشخص است که راه حل حاصل، راه حل درستی نیست. به منظور پیدا کردن جواب بهینه تر تبادل اطلاعات میان گروه‌ها نیاز است.

تبادل اطلاعاتی در بین گروه‌ها نیازمند براداری است که قابل دسترس توسط تمامی گروه‌ها باشد. این بردار، بردار محتوی^۷ نام دارد. برای یک مسئله D بعدی، این بردار نیز D بعد خواهد داشت. زمانیکه گروه j th فعال است، j th سطر از بردار محتوی به وسیله بهترین ذره گروه j ام و سایر عناصر این بردار توسط $gbest$ های سایر گروه‌ها ($D-1$ گروهی که در زمان ارزیابی گروه نام ثابت باقی می‌مانند) پر می‌شود. در هر لحظه این بردار به شکل $b(j,z)$ است. امتیاز این الگوریتم این است که تابع خطا f ، بعد از به روز رسانی هر عضو از بردار محتوی ارزیابی می‌شود. ارزیابی بردار محتوی برای هر گروه به شکل زیر است:

$$\begin{aligned} \text{If } f(b(j, P_{j,x_i})) &< f(b(j, P_{j,pbest_i})) \\ P_{j,pbest_i} &= P_{j,x_i} \\ \text{If } f(b(j, P_{j,pbest_i})) &> f(b(j, P_{j,pbest_i})) \end{aligned} \quad (4)$$

⁷ Context Vector

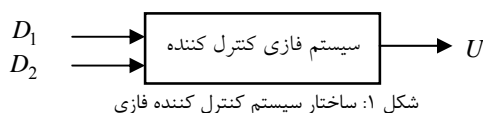
$$P_{j,gbest_i} = P_{j,pbest_i} \quad (5)$$

نتیجه روابط (۴) و (۵) باعث رسیدن به شرایط معتبرتر می‌شود. با توجه به اینکه تمامی ذرات در تشکیل بردار b ، به عنوان بردار محتوی شرکت دارند، تعامل و تبادل اطلاعات بین ذرات گروه‌ها به وجود می‌آید. الگوریتم CPSO دارای سرعت همگرایی بالایی است. اما همچنان مشکل قرار گرفتن در بهینه‌های محلی با توجه به، به روز شدن یک گروه در هر لحظه، پا برجا است. ترکیب این الگوریتم با PSO استاندارد، به نحوی که بهترین ویژگی‌های هر دو الگوریتم را داشته باشد (سرعت همگرایی بالا و فرار از مینیمم‌های محلی)، تحت عنوان CPSO-H معرفی شده است [11]. در این الگوریتم در جایگاه CPSO در دام مینیمم‌های محلی قرار می‌گیرد، الگوریتم PSO شروع به کار می‌کند. اما تشخیص اینکه در هر لحظه کدام الگوریتم شروع به کار کند، سخت و دشوار است.

راه ساده‌تر، جای دادن این دو الگوریتم درون یکدیگر است. بعد از یکبار تکرار الگوریتم CPSO، بردار محتوی به دست آمده به عنوان ورودی الگوریتم PSO در نظر گرفته می‌شود. برای محاسبه بهینه‌ترین ذره در این مرحله، از بردار محتوی حاصل از الگوریتم CPSO استفاده می‌شود. برای تمامی ذرات جمعیت، الگوریتم PSO اجرا می‌شود و راه‌حل حاصل از اجرای الگوریتم PSO، بر اساس تعداد ابعاد هر زیرگروه تکه تکه شده و به عنوان بردار محتوی به الگوریتم CPSO باز گردانده می‌شود. نتیجه به کارگیری و ترکیب این دو الگوریتم، افزایش سرعت الگوریتم و کاهش احتمال قرار گرفتن در مینیمم‌های محلی است. در حالیکه تداوم در جابه‌جایی ذرات باعث کاهش تنوع در جمعیت می‌شود. اگر ذره‌ای که از الگوریتم PSO وارد الگوریتم CPSO می‌شود از سایر ذرات بهینه‌تر باشد، سایر ذرات به سمت آن کشیده می‌شوند و تنوع گروه به مخاطره می‌افتد. یک مکانیزم ساده به منظور جلوگیری از کاهش احتمالی تنوع در گروه‌ها، کاهش تعداد ذراتی است که به طور فعال در جابه‌جایی اطلاعات مشارکت دارند. در [11] نشان داده شده است که CPSO-H کارایی بهتری نسبت به CPSO دارد.

۴- مدل پیشنهادی FCPSO-H

الگوریتم FCPSO-H، ترکیبی از بهینه‌سازی گروه ذرات CPSO-H و منطق فازی است. همانند مدل بهینه سازی گروه ذرات، دارای n ذره است که هر یک از ذرات دارای یک بردار موقعیت و یک بردار سرعت است. فضای جستجویی که ذرات در آن قرار دارند D بعدی است. در الگوریتم پیشنهادی به جای D زیرگروه تک بعدی، زیرگروه‌هایی در نظر گرفته می‌شوند که بیشتر از یک بعد دارند.



به منظور ایجاد پویایی و بالا بردن قابلیت تطبیق با تغییرات محیطی، یک سیستم فازی کنترل کننده پیشنهاد شده است. همانطور که در شکل ۱ نشان داده شده است این سیستم دارای دو ورودی و یک خروجی است. ورودی‌ها به وسیله بهینه‌ترین ذره هر گروه $P_{j,gbest}$ و بهینه‌ترین تجربه هر ذره در گروه $P_{j,pbest_i}$ و طبق فرمول‌های (۶) و (۷) حاصل می‌شود.

$$D1_{j,i} = P_{j,pbest_i} - x_i \quad (6)$$

$$D2_{j,i} = P_{j,gbest} - x_i \quad (7)$$

خروجی این سیستم ضرایب C_k است. استراتژی برنامه‌نویسی ایستا در طی حجم زیادی از آزمایشات ضرایب C_k مناسبی را بدست می‌آورد. اگر چه این الگوریتم‌ها کارایی همگرایی به جواب بهینه را نسبت به الگوریتم PSO استاندارد بهبود می‌بخشد، اما ضرایب C_k قادر به تطبیق خود با تغییرات محیط نیستند [13]. از این جهت توانایی الگوریتم در تطبیق با محیط پویا ضعیف و همگرایی به جواب بهینه را تا حدی محدود می‌شود. بر اساس ورودی‌های سیستم فازی، ضرایب C_1 و C_2 به عنوان خروجی، تحت تاثیر حرکت ذرات و در واقع بهینه عمومی و بهینه‌های محلی هر گروه (با توجه به الگوریتم CPSO-H) هستند. از طرفی این بهینه‌ها در هر مرحله از اجرای الگوریتم و با توجه به تغییرات محیط تغییر می‌کنند. در نتیجه الگوریتم با کمک این ضرایب، خود را با تغییرات محیط تطبیق می‌دهد.

با توجه به اینکه کاهش C_1 و C_2 باعث افزایش جستجوی سراسری و افزایش آن باعث افزایش جستجوی محلی می‌باشد، نمودار C_1 و C_2 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 می‌باشد. قوانین ارائه شده به طور واضح مبنای تصمیم‌گیری را مشخص می‌کند. تعداد قوانین فازی در سیستم

پیشنهادی ۴۹ عدد است. مجموعه قوانین فازی در پروسه فازی کردن ورودی‌ها و دی‌فازی کردن خروجی یکسان انتخاب و تعریف شده است :

$$(H_n, M_n, L_n, Z, L_p, M_p, H_p)$$

که به ترتیب و از چپ به راست "منفی بزرگ"، "منفی میانه"، "منفی کوچک"، "صفر"، "مثبت بزرگ"، "مثبت میانه"، "مثبت کوچک" تعریف شده‌اند.

جدول ۱: قوانین فازی ساختار فازی کنترل کننده

$E1/E2$	HN	MN	LN	Z	LP	MP	HP
HN	LP	LP	MN	LN	LN	Z	LP
MN	MP	LP	LN	LN	LN	Z	LP
LN	HP	HP	Z	MN	Z	LP	MP
Z	HP	HP	MP	Z	MP	HP	HP
LP	HP	LP	Z	MN	LP	LP	HP
MP	MP	LP	Z	LN	Z	MP	HP
HP	LP	Z	LN	LN	LN	HP	HP

مراحل الگوریتم پیشنهادی FCP SO-H به شکل زیر است:

- ۱- گروه ذرات بر اساس الگوریتم CPSO-H مقداردهی اولیه می‌شوند.
 - ۲- ارزیابی ارزش هر ذره با توجه به تابع شایستگی مورد آزمایش.
 - ۳- محاسبه کردن بهینه‌ترین ذره هر گروه $P_j.gbest$ و بهینه‌ترین تجربه هر ذره در گروه $P_j.pbest_i$.
 - ۴- مشخص کردن مقادیر D_1 و D_2 و وارد کردن منطق فازی در الگوریتم (اعمال قوانین فازی).
 - ۵- بدست آوردن نرخ تغییرات ضرایب C_1 و C_2 با استفاده از دی‌فازی کردن.
 - ۶- در ادامه سرعت و مکان هر ذره در صورت بهینه‌تر شدن ذره (با توجه به ارزیابی تابع شایستگی مورد بررسی) به روز می‌شوند.
 - ۷- بررسی شرط خاتمه. (یا همگرا شدن به جواب و یا رسیدن به یک تعداد مشخصی از تکرار)
- باید به این نکته توجه شود که با توجه به اعمال سیستم فازی کنترل کننده به الگوریتم CPSO-H این مراحل در هر یک از گروه‌ها با ابعاد کوچکتر انجام می‌پذیرد.

۵- نتایج شبیه‌سازی‌ها

برای مقایسه الگوریتم FCP SO-H با الگوریتم‌های Fuzzy PSO، CPSO-H و PSO که در بخش‌های سوم و دوم توضیح داده شده، شبیه‌سازی‌های زیادی بر روی هشت تابع استاندارد صورت گرفته است که معمولاً به عنوان معیار سنجش الگوریتم‌های بهینه سازی مورد استفاده قرار می‌گیرند. توابع استفاده شده به همراه محدوده مقداردهی اولیه ذرات و محدوده فضای جستجوی در جدول ۲ مشخص شده است. فضای قابل قبول برای توابع تست به همراه نمودارشان نیز در شکل ۲ مشخص گردیده است. آزمایش بر روی ۸ تابع تست و در دو قسمت ۱۰ بعدی و ۳۰ بعدی انجام شده است. نتایج ارائه شده در شکل ۳ و ۴ میانگین ۳۰ بار اجرای الگوریتم‌ها می‌باشد. در جدول ۳ تنظیمات مربوط به متغیرهای ورودی فازی $DL_{j,i}$ و $D2_{j,i}$ و متغیرهای خروجی C_1 و C_2 برای تابع Rosenbrock آمده است. لازم به ذکر است که این محدوده‌ها با توجه نوع توابع تغییر پیدا می‌کند. این تغییرات با توجه به محدوده فضای جستجو و محدوده مقداردهی اولیه ذرات انجام می‌گیرد.

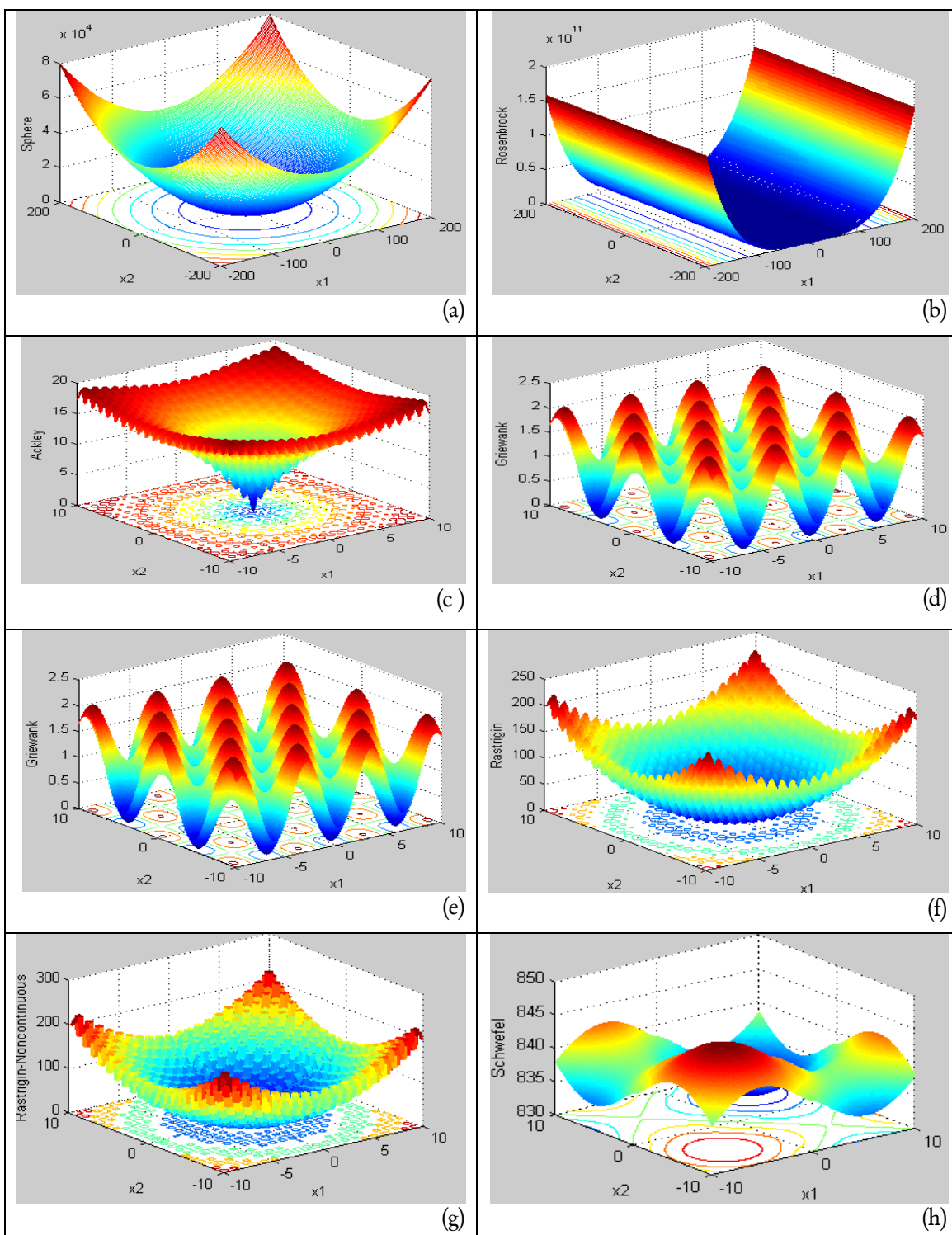
نتایج ارائه شده در جدول ۴ و ۵ به ترتیب در فضای جستجوی ۱۰ بعدی و ۳۰ بعدی، برتری الگوریتم پیشنهادی را در برابر روش‌های موجود نشان می‌دهد. از مقایسه این نتایج می‌توان نتیجه گرفت که رفتار تابع پیشنهادی خیلی بهتر از سه الگوریتم PSO و CPSO-H و Fuzzy PSO می‌باشد.

جدول ۲: توابع تست و مقادیر اولیه برای توابع تست

Name	F	X*	Search Range	Initiation Range
Sphere	$f_1(x) = \sum_{i=1}^D x_i^2$	0,0,...,0]	[-100,100] ^D	[-100,50] ^D
Rosenbrock	$f_2(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1}^2)^2 + (x_i^2 - 1)^2)$	[-1,1],...,[-1,1]	[-2.048,2.048] ^D	[-2.048,2.048] ^D
Ackley	$f_3(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e$	[-30,30],...,[-30,30]	[-32.76,32.76] ^D	[-32.768,32.768] ^D
Griewank	$f_4(x) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[-60,60],...,[-60,60]	[-600,600] ^D	[-600,600] ^D
Weierstrass	$f_5(x) = \sum_{i=1}^D \left(\sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k (x_i + 0.5))] \right) - D \sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k 0.5)]$ $a = 0.5, b = 3, k_{\max} = 20$	[-0.5,0.5],...,[-0.5,0.5]	[-0.5,0.5] ^D	[-0.5,0.5] ^D
Rastrigin	$f_6(x) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10)$	[-5.12,5.12],...,[-5.12,5.12]	[-5.12,5.12] ^D	[-5.12,5.12] ^D
Noncontinuous Rastrigin	$f_7(x) = \sum_{i=1}^D (y_i^2 - 10 \cos(2\pi y_i) + 10)$ $y_i = \begin{cases} x_i & x_i < 1/2 \\ \text{round}(2x_i)/2 & x_i \geq 1/2 \end{cases}$ $\text{for } i = 1, 2, \dots, D$	[-5.12,5.12],...,[-5.12,5.12]	[-5.12,5.12] ^D	[-5.12,5.12] ^D
Schwefel	$f_8(x) = 418.9829 * D - \sum_{i=1}^D x_i \sin\left(x_i ^{\frac{1}{2}}\right)$	[-20.96,20.96],...,[-20.96,20.96]	[-500,500] ^D	[-500,500] ^D

جدول ۳: محدوده تنظیمات متغیرهای فازی

Function	Fuzzy Variable	H _n	M _n	L _n	Z	L _p	M _p	H _p
	D1 _{j,i}	(-100,-50)	(-60,-20)	(-30,5)	(-5,5)	(-5,30)	(20,60)	(50,100)
f ₁	D2 _{j,i}	(-100,-50)	(-60,-20)	(-30,5)	(-5,5)	(-5,30)	(20,60)	(50,100)
	$\Delta C_1 / C_2$	(-0.1,-0.04)	(-0.045,-0.015)	(-0.025,0.005)	(-0.005,0.005)	(-0.005,0.025)	(0.015,0.045)	(0.04,0.1)



شکل ۲: نمودار ۸ تابع (a) Sphere، (b) Rosenbrock، (c) Ackley، (d) Griewank، (e) Weierstrass، (f) Rastrigin، (g) Noncontinuous Rastrigin، (h) Schwefel

جدول ۴: مقایسه الگوریتم FCPHO-H و الگوریتم PSO و CPSO-H و FPSO در ۱۰ بعد

	F1	F2	F3	F4
<i>Original PSO</i>	61.6 ± 207.12	68.824 ± 97.387	0.7199 ± 0.019495	1.5058 ± 0.0026229
<i>Fuzzy PSO</i>	3.3297e-314 ± 0	0.11752 ± 1.2629	2.1316e-014 ± 1.7895e-028	0.59075 ± 0.0015774
<i>Cooperative PSO/CPSO-H</i>	1.0872e-311 ± 0	3.9871 ± 2.8252	7.1054e-015 ± 5.9603e-029	0.23848 ± 0.0025699
<i>Fuzzy CPSO-H</i>	9.1231e-214 ± 0	2.5446 ± 1.2453	3.5527e-015 ± 0	0.012316 ± 0.00011212
	F5	F6	F7	F8
<i>Original PSO</i>	8.8185 ± 0.39671	6.2292 ± 1.2521	34.066 ± 15.034	4151.1 ± 0.011978
<i>Fuzzy PSO</i>	0.52964 ± 0.13468	0 ± 2.9801e-030	8.8818e-015 ± 2.1778	4150.8 ± 0.001993
<i>Cooperative PSO/CPSO-H</i>	0.0021739 ± 0.078251	0 ± 1.7744e-028	1 ± 0.23333	4150.4 ± 2.6164e-017
<i>Fuzzy CPSO-H</i>	0.0039687 ± 1.4817e-006	0 ± 0	1 ± 0.44444	4150.3 ± 9.1909e-026

همانطور که در جدول ۴ دیده می‌شود، با توجه به اینکه تعداد ابعاد پایین است الگوریتم Fuzzy PSO در برخی از توابع نظیر F1، F2 و F7 به نتایج بهتری رسیده است. در سایر توابع نیز به جواب‌های قابل قبولی رسیده است. روند خطی تغییرات در نمودارهای الگوریتم PSO نشان دهنده قرار گرفتن این الگوریتم در بهینه‌های محلی است. الگوریتم CPSO با توجه به کوچک بودن ابعاد به نتایجی بدتر از Fuzzy PSO دست پیدا کرده است. البته لازم به ذکر است که در رسیدن به این جواب‌ها دارای سرعت بالاتری نسبت به Fuzzy PSO می‌باشد. در کورد الگوریتم پیشنهادی باید گفت که در تابع F6 به عدد صفر رسیده است. در تابع F3 و F4 هم به بهترین جواب رسیده است.

البته خیلی انتظار رسیدن به جواب‌های بهتری نسبت به Fuzzy PSO نمی‌رفت. زیرا تعداد ابعاد در این آزمایش پایین است و این امر باعث شده تا این الگوریتم در برخی از توابع به جواب‌های بهتری برسد.

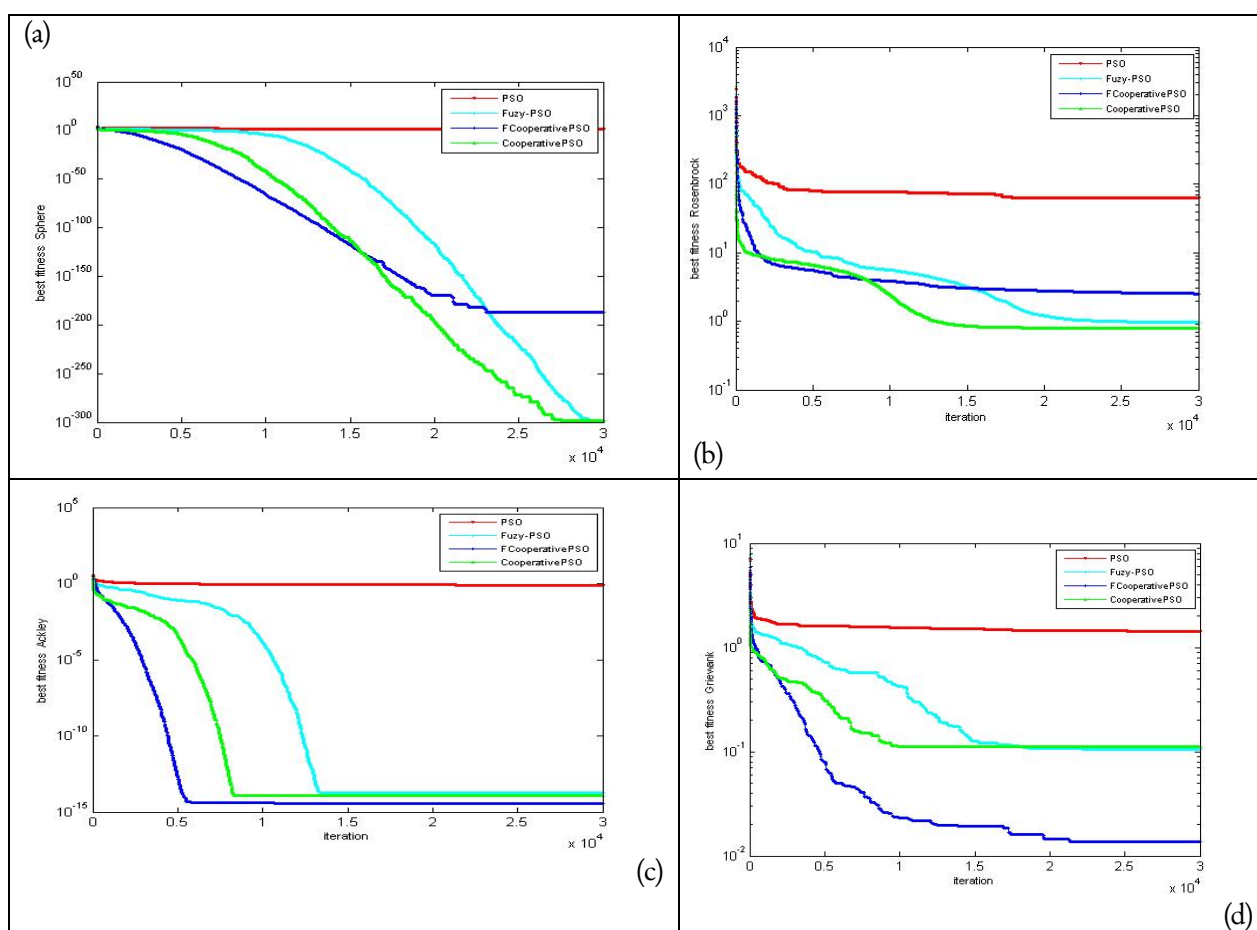
جدول ۵: مقایسه الگوریتم FCPHO-H و الگوریتم PSO و CPSO-H و FPSO در ۳۰ بعد

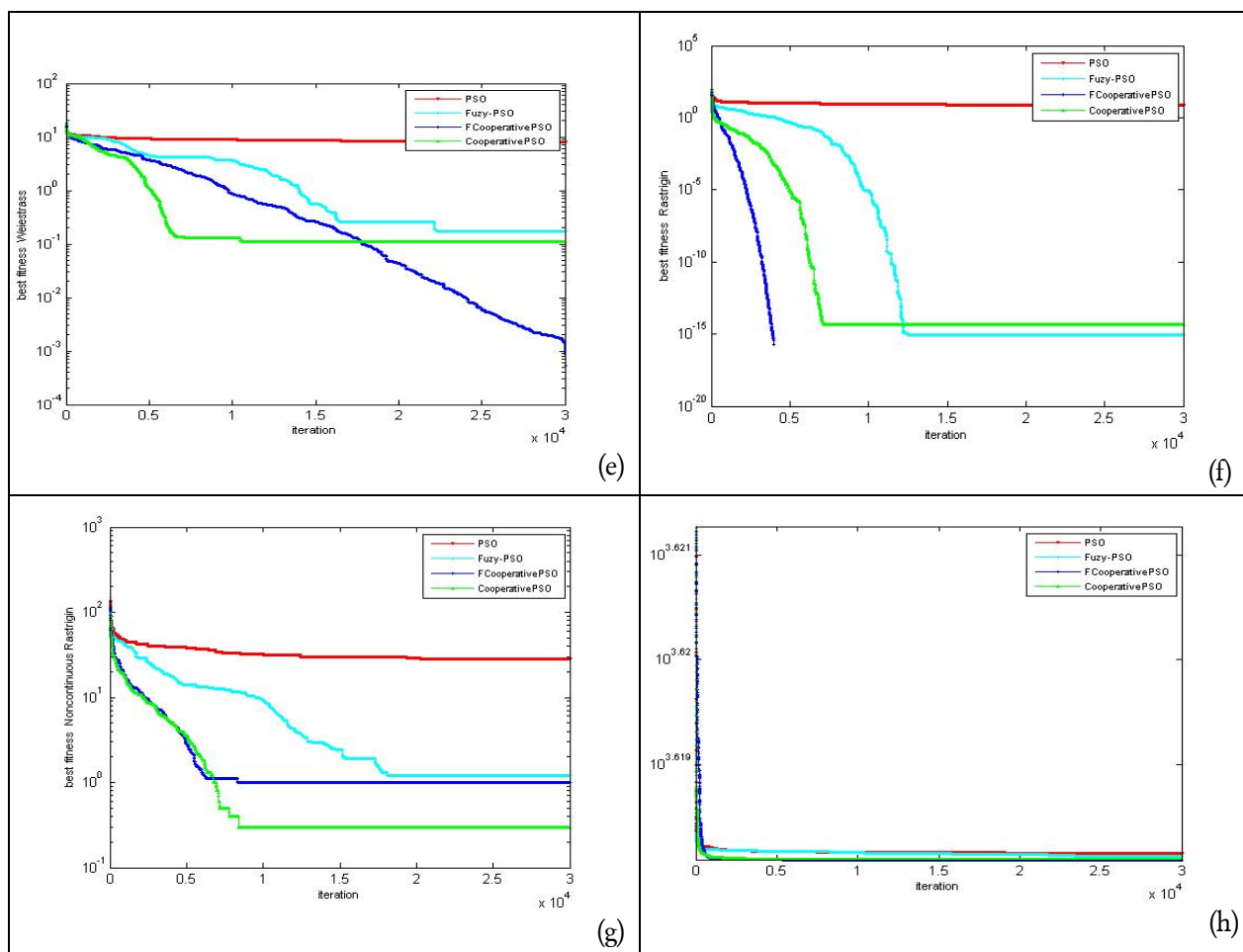
Algorithm	F1	F2	F3	F4
<i>Original PSO</i>	941.31 ± 13119	1610.6 ± 7127.1	2.1537 ± 0.012453	7.3611 ± 0.30691
<i>Fuzzy PSO</i>	1.1967e-040 ± 8.7323e-072	77.241 ± 567.57	3.4204e-010 ± 3.3604e-020	0.022151 ± 0.0007942
<i>Cooperative PSO/CPSO-H</i>	4.7755e-033 ± 1.2981e-048	23.7 ± 6.7224	1.676e-007 ± 4.5419e-014	0.031942 ± 0.00046773
<i>Fuzzy CPSO-H</i>	3.0918e-077 ± 1.8117e-1134	14.574 ± 21.261	6.3949e-014 ± 2.5819e-028	0 ± 3.6682e-005
Algorithm	F5	F6	F7	F8
<i>Original PSO</i>	36.845 ± 0.82859	109.4 ± 40.833	291.79 ± 184.2	12456 ± 0.21194
<i>Fuzzy PSO</i>	1.2692 ± 0.76157	6.2172e-012 ± 1.2733e-023	50 ± 1154.2	12454 ± 0.13615
<i>Cooperative PSO/CPSO-H</i>	1.4738 ± 4.975	1.5987e-014 ± 5.981e-018	25 ± 367.34	12451 ± 0.0010138

<i>Fuzzy CPSO-H</i>	0.031958 ± 0.026626	$0 \pm 5.6097e-031$	9 ± 11.556	$12451 \pm 9.8048e-022$
---------------------	-------------------------	---------------------	----------------	-------------------------

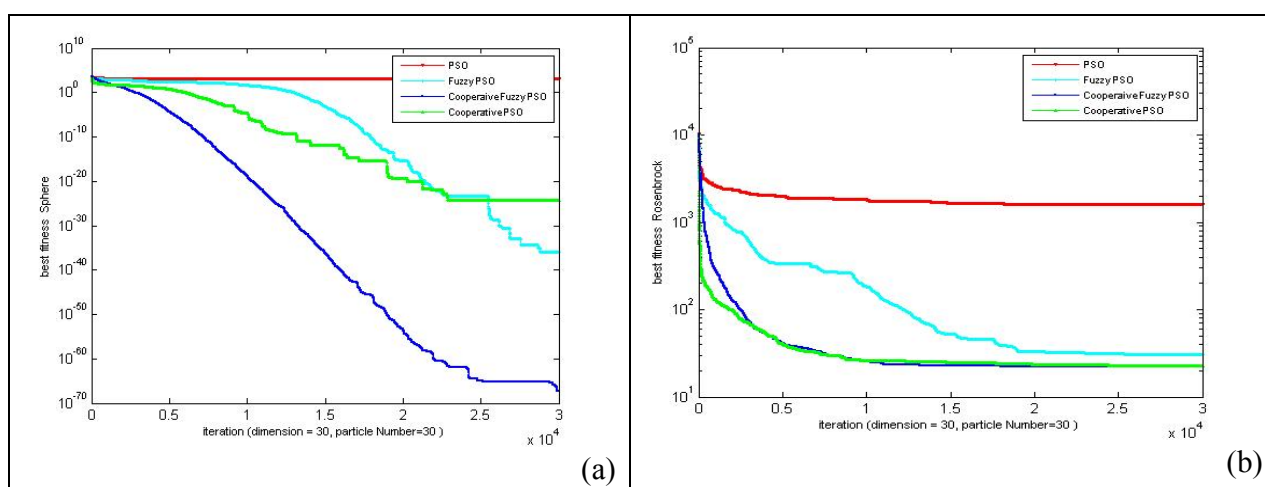
همانطور که انتظار می‌رفت با بالا رفتن ابعاد فضای جستجو الگوریتم پیشنهادی به جواب‌های بهینه‌تری در تمامی توابع رسیده است. الگوریتم پیشنهادی در توابع F4 و F6 به عدد صفر رسیده است. و همانطور که در نمودارها مشخص است از سرعت بالای الگوریتم CPSO و کارایی الگوریتم Fuzzy PSO استفاده کرده و هم سرعت بالاتری نسبت به سایر روش‌ها دارد و هم به جواب‌های بهتری رسیده است.

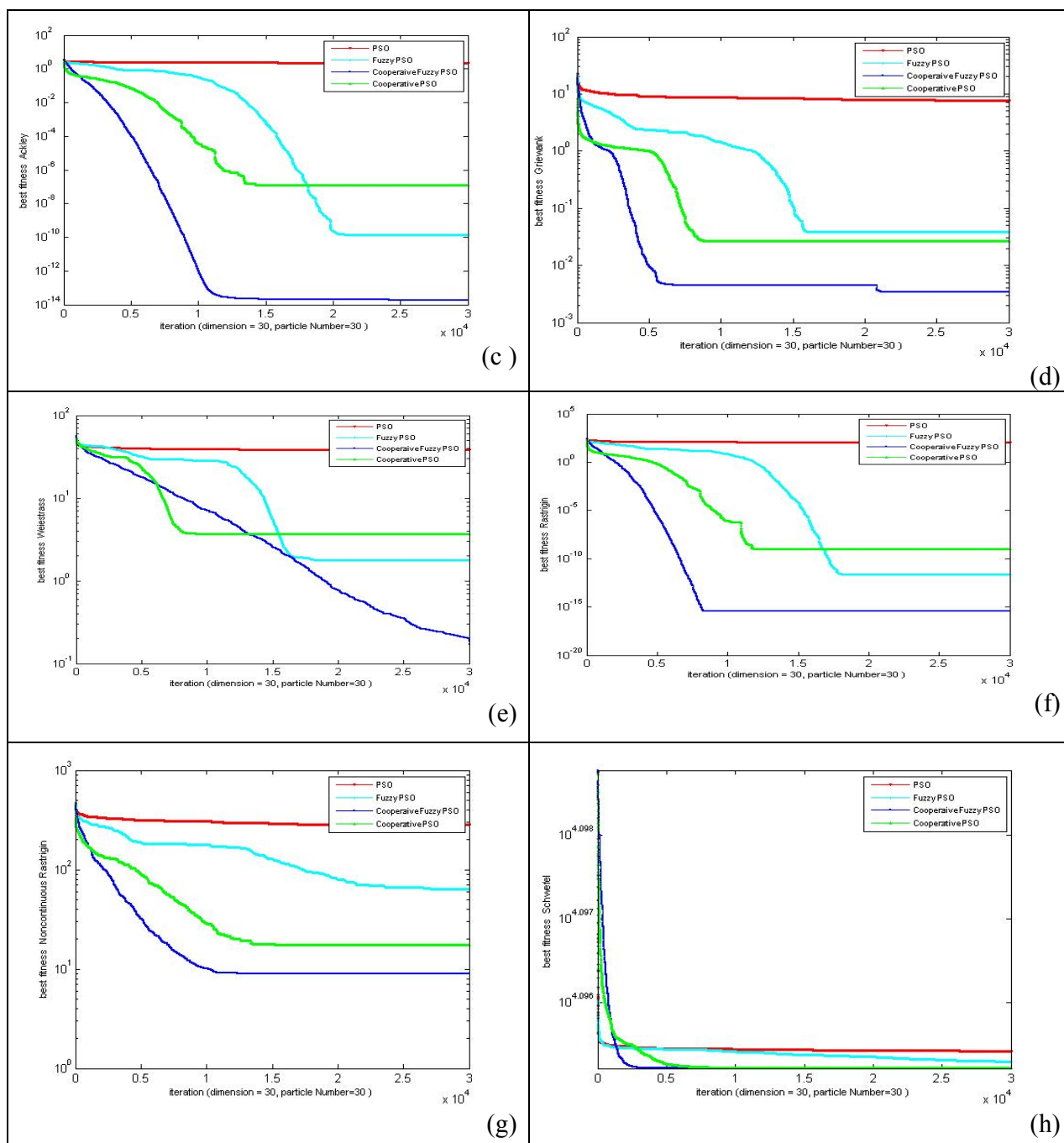
در این آزمایش الگوریتم FuzzyPSO همچنان در بعضی از توابع به نتایج بهتری نسبت به CPSO دست یافته است اما سرعت پایین‌تری دارد.





شکل ۳: میانگین بهترین تابع شایستگی ذرات نسبت به تعداد تکرارهای الگوریتم در ۲۰ تکرار، ۳۰ ذره، ۱۰ بعد، ۳۰۰۰۰ تکرار ۸ تابع (a) Sphere Rosenbrock (b)، Ackley (c)، Greiwank (d)، Weierstrass (e)، Rastrigin (f)، Noncontinuous Rastrigin (g) و Schwefel (h)





شکل ۴: میانگین بهترین تابع شایستگی ذرات نسبت به تعداد تکرارهای الگوریتم در ۲۰ تکرار، ۳۰ ذره، ۳۰ بعد، ۳۰۰۰۰ تکرار ۸ تابع (a) Rosenbrock (b), Sphere Schwefel(h), Noncontinuous Rastrigin(g), Rastrigin (f) , Weierstrass (e) , Griewank (d) , Ackley (c).

۶ نتیجه گیری

در این فصل یک الگوریتم بهینه‌سازی جدید به نام FCPSO-H که ترکیبی از منطق فازی و نوع خاصی از بهینه‌سازی گروه ذرات به نام CPSO-H می‌باشد، ارائه گردید. با توجه به نتایج بدست آمده، می‌توان گفت که استفاده از منطق فازی این امکان را برای الگوریتم PSO استاندارد فراهم می‌کند تا در مقابل تغییرات محیطی، عکس‌العمل مناسبی نشان دهد. و بتواند خود را با تغییرات محیط تطبیق دهد. از طرفی با استفاده از یادگیری مشارکتی در الگوریتم CPSO-H سرعت همگرایی افزایش و مشکل قرار گرفتن در بهینه‌های محلی را تا حدودی حل شده است. در مجموع با استفاده از ویژگی‌های مثبت این دو الگوریتم توانسته به جواب‌های قابل قبولی دست پیدا کند.

کارایی این الگوریتم در فضای جستجو با ابعاد بالا، بیشتر است، هرچند که در فضای جستجو با ابعاد پایین نیز به نتایج خوبی دست پیدا کرده است.

۷ مراجع

- [1] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization", Proceedings of IEEE International Conference on Neural Networks, Piscataway, pp. 1942-1948, 1995.
- [2] J. Kennedy, S. Worlds and M. Minds, "Effects of Neighborhood Topology on Particle Swarm Performance", Proceedings of the IEEE Congress on Evolutionary Computation, Vol. 3, pp. 1931-1938, July 1999.
- [3] F. V. Bergh and A. P. Engelbrecht, "Effects of Swarm Size on Cooperative Particle Swarm Optimizers", Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, USA, pp. 892-899, 2001.
- [4] M. Løvberg, T. Rasmussen and T. Krink, "Hybrid Particle Swarm Optimizer with Breeding and Subpopulation", Proceedings of the Third Genetic and Evolutionary Computation Conference, Vol. 1, pp. 469-476, 2001.
- [5] A. M. Abdelbar, S. Adbelshahid and D. C. Wunsch, "Fuzzy PSO: A Generalization of Particle Swarm Optimization", Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, July, 2005.
- [6] H. Liu and A. Abraham, "A Hybrid Fuzzy Variable Neighborhood Particle Swarm Optimization Algorithm for Solving Quadratic Assignment Problems", Journal of Universal Computer Science, Vol. 13, No. 7 pp. 1032-1054 2007.
- [7] S. H. Zahir and S. A. Seyedin, "Swarm Intelligence Based Classifiers", Journal of the Franklin Institute 344, pp. 362-376, 2007.
- [8] H. Liu and A. Abraham, "A Fuzzy Turbulent Particle Swarm Optimization", Int. J. Innovative Computing and Applications, vol. 1, no. 1, 2007.
- [9] J. J. Liang and A. K. Qin, "Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions", IEEE Transactions on Evolutionary Computation, vol. 10, no. 3, June 2006.
- [10] F. V. Berg and A. P. Engelbrecht, "A Cooperative Approach to Particle Swarm Optimization", IEEE Transactions on Evolutionary Computation, Vol. 8, Nno. 3, June 2004.
- [11] F. V. Bergh, "A Analysis of Particle Swarm Optimizers", Survey, November 2001.
- [12] R. Rastegar, M. R. Meybodi and K. Badie, "A New Discrete Binary Particle Swarm Optimization based on Learning Automata", Proceedings of International Conference on Machine Learning and Applications (ICMLA2004).
- [13] A. Ratnaweera, S. Halgamuge, and H. Watson, "Particle Swarm Optimization with Self-Adaptive Acceleration Coefficients", Proceedings of the First International Conference on Fuzzy Systems and Knowledge Discovery, pp. 2411-2418, December 2003.
- [14] J. Kennedy and R. C. Eberhart, "A Discrete Binary Version of the Particle Swarm Algorithm", Conference on System and Cybernetics, pp. 4104-4109, 1997.
- [15] M. H. Noroozibeyrami and M. R. Meybodi, "Improving Particle Swarm Optimization using Fuzzy Logic", Proceedings of the Second Iranian Data Mining Conference, Amir Kabir University of Technology, Tehran, Iran, Sept. 21-22, 2008.
- [16] M. H. Noroozibeyrami and M. R. Meybodi, "Cooperative Fuzzy Particle Swarm Optimization", Proceedings of the 2nd Joint Congress on Fuzzy and Intelligent Systems, Malek Ashtar University of Technology, Tehran, Iran, 28-30 October, 2008.
- [17] Y. Zheng, L. Ma, L. Zhang and Qian, "Empirical Study of Particle Swarm Optimizer with Increasing Inertia Weight", In Proceeding of IEEE Congress on Evolutionary Computation, Pages221-226. IEEE Press, 2003.
- [18] B. Niu, Y. Zhu, X. He and H. Wu, "MCP SO: A Multi-Swarm Cooperative Particle Swarm Optimizer", Elsevier Mathematics and Computation, No. 185, pp. 1050-1062, 2007.