طبقه‌بندی موجودی با استفاده از بهینه‌سازی جمعی راه‌حل‌های چندهدفه

منصور اسماعیل‌زاده [۱]
امین حسین‌پور [۲]
محمدرضا نامدار [۳]

چکیده

طبقه‌بندی موجودی یکی از تکنیک‌های مهم در حوزه‌ی مدیریت موجودی است. به‌دلیل تنوع و حجم بالای اقلام موجودی در یک شرکت، مدیران ناگزیر هستند آنها را طبقه‌بندی کنند. بنابراین، بخشی از تلاش پژوهشگران به‌منظور یافتن روش‌هایی بوده است که با تعیین تعداد طبقات موجودی، توان کنترل مدیریت را افزایش دهند. در این مقاله، از الگوریتم بهینه‌سازی جمعی راه‌حل‌های چندهدفه استفاده می‌شود. این الگوریتم از سوی «چی یانگ» و «سو ویی» در سال ۲۰۰۸ ارائه شده است.[۴] الگوریتم بهینه‌سازی جمعی راه‌حل‌های چندهدفه نوعی الگوریتم تکاملی است که چندهدفه‌بودن تابع آن، به مدیریت این امکان را می‌دهد تا هم‌زمان به‌دنبال بهینه‌سازی اهداف متعددی باشد. حداقل‌کردن هزینه‌های نگهداری و سفارش‌دهی و حداکثرکردن نرخ گردش موجودی در این مدل مد نظر هستند. پس از نوشتن برنامه‌ی نرم‌افزاری مدل، آن را روی یک نمونه‌ی ۱۰۰تایی مورد آزمون قرار دادیم. نتایج نشان می‌دهد که این الگوریتم، می‌تواند هزینه‌های نگهداری و سفارش‌دهی را به‌طور قابل ملاحظه‌ای کاهش داده و نرخ گردش موجودی را افزایش دهد.

کلمات کلیدی: بهینه‌سازی مسائل چندهدفه[۵]، الگوریتم بهینه‌سازی جمعی راه‌حل‌های چندهدفه[۶]، طبقه‌بندی موجودی[۷].

۱- عضو هیئت علمی دانشگاه ولی‌عصر(عج) رفسنجان(نویسنده‌ی مسئول) Esmailzadeh@vru.ac.ir
۲- دانشجوی کارشناسی ارشد مدیریت بازرگانی دانشگاه ایلام
۳- دانشجوی کارشناسی مدیریت صنعتی دانشگاه ولی‌عصر(عج) رفسنجان
۴- برای کسب اطلاعات بیشتر به منبع شماره‌ی ۱۵ مراجعه شود.
۵- Multi Objective Optimization Problems.
۶- Multi Objective Particle Swarm Optimization Algorithm.
۷- Inventory Classification.

تاریخ دریافت : ۹۱/۲/۱۸ تاریخ پذیرش : ۹۲/۳/۱۲

مقدمه

در عرصه‌ی جهانی، «رقابت» و «جنگ برای بقا» دو واژه‌ای هستند که ذهن مدیران شرکت‌ها را به‌خود مشغول کرده‌اند. از سوی دیگر واژه‌ی بقا این مفهوم را به ذهن متبادر می‌کند که هدف تنها زنده‌ماندن است، ولی در عمل این‌گونه نیست، زیرا شرکت‌ها در کنار بقا به‌دنبال کسب سود و سهم بیشتر در بازار هستند. از این‌رو باید به‌دنبال راه‌هایی باشند تا مشتریان فعلی خود را همیشه راضی نگه داشته و درعین‌حال مشتریان بالقوه را نیز جذب کنند. در چنین محیطی است که موجودی اهمیت بسیار زیادی پیدا می‌کند. در واقع قابلیت دسترسی به اقلام مدنظر در زمان و مکان مناسب موجب دستیابی شرکت به اهداف خدمت‌دهی به مشتری، بهره‌وری، سود و بازگشت سرمایه می‌شود. تنوع قابل ملاحظه‌ی تقاضا و فعالیت‌ها، مؤسسات تولیدی را مجبور کرده است تا اقلام متنوعی از مواد اولیه و محصولات تولیدی در انبارهای خود داشته باشند. این تنوع اقلام موجودی، شرکت‌ها را ناگزیر به طبقه بندی این اقلام و تدوین سیاست‌های موجودی اثربخش می‌کند[۲و۱]. یک شرکت با انباشت سرمایه به‌صورت موجودی از به‌کارگیری این سرمایه برای دستیابی به اهدافی دیگر، چون خرید تجهیزات یا توسعه‌ی محصولات جدید صرف نظر می‌کند، بنابراین نوعی هزینه‌ی سرمایه‌ای که به‌صورت نرخ بهره‌ی سالیانه بیان می‌شود، بر سرمایه‌گذاری موجودی تحمیل می‌شود[۳].

طرح‌ها و پژوهش‌های بسیاری از سوی پژوهشگران به‌منظور ارائه‌ی روشی مناسب برای اجرای بهترین طبقه‌بندی ممکن انجام شده‌است. پیش از معرفی این پژوهش‌ها لازم است که روش سنتی طبقه‌بندی را معرفی کنیم. طبقه‌بندی کلاسیک (سنتی) ABC از مشهورترین روش‌های طبقه‌بندی اقلام موجودی است[۲]. هدف اصلی از طبقه‌بندی ABC متمرکزسازی تلاش برای کنترل شدید اقلام طبقه‌ی A، کنترل کمتر اقلام طبقه‌ی B و کنترل بسیار کم اقلام طبقه‌ی C است[۴]. در طبقه‌بندی کلاسیک ABC اقلام براساس دو معیار قیمت واحد و مصرف سالیانه یا به‌عبارت دیگر

معیار واحد ارزش مصرف سالیانه طبقه‌بندی می‌شوند[۵].
«کوهن»[۱] و «ارنست»[۲] [۶] از روشی آماری به‌نام تحلیل خوشه‌ای برای طبقه‌بندی موجودی استفاده کردند. برتری اصلی این روش این است که می‌تواند خودش را با مقدار زیادی از ویژگی‌های ترکیبی که برای اهداف راهبردی و عملیاتی مهم هستند، وفق دهد. «پرتوی»[۳] و« بورتن»[۴] [۷] با استفاده از «فرایند تجزیه‌وتحلیل سلسله‌مراتبی»[۵] به طبقه‌بندی چندمعیاره‌ی موجودی پرداختند. اگرچه روش ارائه‌شده از سوی آنها قابلیت کاربرد برای طیف وسیعی از اقلام را دارد، ولی ورود اقلام جدید ممکن است رتبه‌ی اقلام قبلی را تغییر دهد. «فلورس»[۶] و «وای‌بارک»[۷] نخستین کسانی هستند که طبقه‌بندی ABC چندمعیاره را بررسی کردند[۸]. از طرف دیگر الگوریتم‌های اکتشافی[۸] و تکاملی[۹] از دیگر روش‌ها در حوزه‌ی طبقه‌بندی موجودی هستند. «گوونیر»»[۱۰] و «اِرل»[۱۱] [۴] روشی را استفاده کردند که در آن با استفاده از الگوریتم ژنتیک، وزن معیارها با توجه به نقاط برش ABو BC تعیین می‌شود. پرتوی و «آناندراجان»[۱۲] [۹] «شبکه‌ی عصبی مصنوعی»[۱۳] را برای طبقه‌بندی موجودی ارائه دادند. در سال‌های ۱۳۸۸ و ۱۳۹۰ «اسماعیل‌زاده» و همکاران در دو مقاله‌ی جداگانه [۲و۴۶] دو روش کمّی برای مقایسه‌ی روش‌های به‌کاررفته در طبقه‌بندی چندمعیاره‌ی موجودی ارائه دادند.

1.Cohen.
2.Ernst.
3.Partovi.
4.Burton.
5.Analytical Hierarchy process.
6.Flores.
7. Whybark.
8. Heuristic Algorithms.
9. Evolutionary Algorithms (EAs).
10. Guvenir.
11. Erel.
12. Anandarajan.

یکی از جدیدترین الگوریتم‌های ریاضی که می‌تواند به مدیریت موجودی در حوزه‌ی طبقه‌بندی موجودی کمک شایانی کند، الگوریتم بهینه‌سازی جمعی راه‌حل‌ها[1] است. این الگوریتم که از دسته‌ی الگوریتم‌های همگرا[2] می‌باشد، قادر است تا ما را در دستیابی به یک پاسخ مناسب با توجه به چندین هدف یاری کند. از سوی دیگر درنظرگرفتن اهداف متنوع و گاهی متضاد در تصمیم‌گیری یکی از مهم‌ترین دغدغه‌های مدیریت است. این چالش به قلمرو مدیریت موجودی نیز پا گذاشته است، بنابراین مدیر یک شرکت هنگام تصمیم‌گیری در خصوص تعداد طبقات موجودی لازم است به هزینه‌ی نگهداری-که به‌نوعی ارزش ریالی اقلام و اهمیت استراتژیک آنها را نشان می‌دهد- نرخ گردش موجودی و همبستگی تقاضای بین اقلام موجودی توجه کند.

در این مقاله سعی شده است تا با استفاده از الگوریتم «بهینه‌سازی جمعی راه‌حل‌های چندهدفه»[3] کاهش هزینه‌های نگهداری و سفارش‌دهی از یک‌سو و از سوی دیگر افزایش نرخ گردش موجودی اقلام مورد نیاز برای تولید کالای نهایی محقق شود.

خاستگاه بهینه‌سازی جمعی راه‌حل‌ها

بهینه‌سازی جمعی راه‌حل‌ها از رفتارهای اجتماعی دسته‌های پرندگان و گروه‌های ماهی‌ها الهام گرفته شده است و برای نخستین‌بار در سال ۱۹۹۵ از سوی «ابرهارت»[4]-روان‌شناس اجتماع- و «کندی»[5] -مهندس الکترونیک- توسعه داده شد[۱۵-۱۰].

هدف این روش که نوعی الگوریتم بهینه‌سازی تکاملی است، شبیه‌سازی حرکت

1. Particle Swarm Optimization Algorithm (PSOA).
2. Convergence Algorithms.
3. Multi Objective Particle Swarm Optimization (MOPSO).
4. Eberhart.

دلپذیر و رقص‌گونه‌ی گروه پرندگان است و مطالعات آن به‌عنوان بخشی از پژوهش اجتماعی ـ شناختی براساس تفکر «هوش جمعی»[1] در جوامع زیستی صورت گرفته است[۱۴]. این روش همچون روش‌های الگوریتم تکاملی استاندارد[2] و الگوریتم ژنتیک و استراتژی‌های تکاملی[3] از طبیعت الهام گرفته شده، ولی برخلاف تکامل بر پایه‌ی رفتار گروهی و تقلیدی پرندگان و ماهی‌ها استوار است[۱۶]. این روش مزیتی تکاملی را برای مسائل بهینه‌سازی پیچیده ارائه می‌کند و در آن همه‌ی راه‌حل‌ها به کشفیات و تجارب راه‌حل‌های مجاور خود در جست‌وجوی بهترین راه‌حل وابسته‌اند[۲۳-۱۷].

بهینه‌سازی جمعی راه‌حل‌ها از دو منشأ گرفته می‌شود. به‌طور اعم این الگوریتم از یک‌سو با زندگی مصنوعی، گروه پرندگان، دسته‌ی ماهی‌ها و تئوری جمعی مرتبط است که بسیار بدیهی هستند و از سوی دیگر به‌طور ویژه با محاسبات تکاملی از قبیل الگوریتم ژنتیک[۲۴] و برنامه‌ریزی تکاملی[۲۵] گره خورده است.

دانشمندان متعددی شبیه‌سازی کامپیوتری متنوعی را به‌منظور تفسیر حرکت ارگانیسم در گروه پرندگان یا دسته‌ی ماهی‌ها انجام داده‌اند. به‌ویژه «رینولدز»[4] [۲۶]، «هپنر»[5] و «گراناندر»[6] [۲۷] روی شبیه‌سازی گروه پرندگان کار کرده‌اند. هر دو مدل ارائه‌شده از سوی رینولدز و هپنر به‌شدت بر تغییر فواصل بین فردی استوارند؛ یعنی فرض می‌شود که رفتار گروهی تابعی از تلاش پرندگان برای حفظ فاصله‌ی بین خود و دیگر اطرافیانشان است. «ویلسون»[7]- زیست‌شناس اجتماعی-[۲۸] فرض کرد که اعضای هر دسته می‌توانند از مزیت برخورداری از اکتشافات و تجارب دیگر اعضای

1. Swarm Intelligence.
2. Standard Evolutionary Algorithm (SEA).
3. Evolutionary Strategies(ESs)
4. Reynolds
5. Heppner
6. Granander

گروه در طول جست‌وجو برای غذا بهره‌مند شوند. همچنین وی فرض کرد که وجود این برتری با توجه به رقابت برای غذا قطعی است و این در صورتی است که منابع به‌طور غیرقابل پیش‌بینی توزیع شده باشند. با توجه به این نکته، به اشتراک‌گذاری اطلاعات بین اعضاء گونه‌های مشابه نوعی برتری تکاملی بوده و برای توسعه‌ی الگوریتم بهینه‌سازی جمعی راه‌حلی اساسی است.

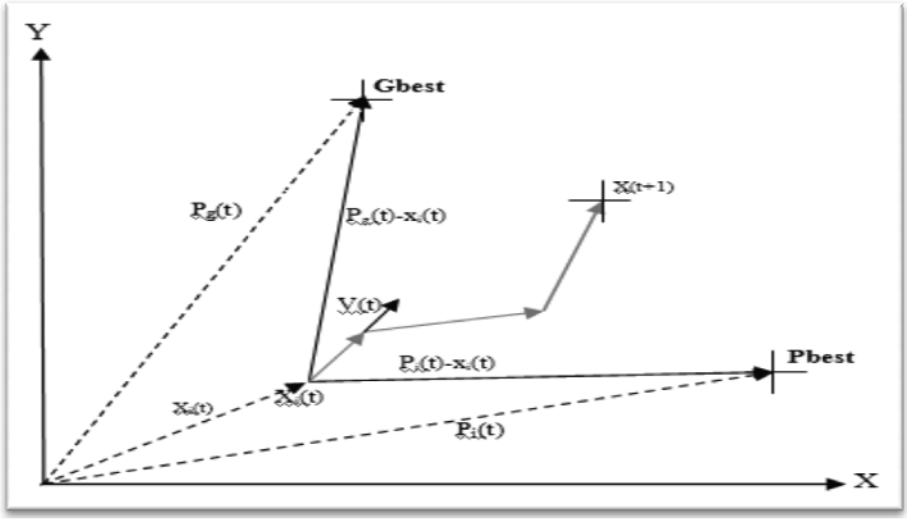
الگوریتم بهینه‌سازی جمعی راه‌حل‌ها

در دسته‌های ماهی یا گروه پرندگان، هر فرد یا آگاهی اندکی دارد یا اصلاً آگاهی ندارد؛ یعنی مانند انسان‌ها هوشمند عمل نمی‌شود. ولی این موجودات غیرهوشمند می‌توانند با تعامل با دیگر اعضا یا گردش در محیط، کارهای بسیار پیچیده‌ای انجام دهند. این الگوریتم را می‌توان به‌خوبی از طریق تشریح وضعیت فرضی زیر توضیح داد:

گروهی از پرندگان در فضایی در حال پروازند و به‌دنبال غذا می‌گردند، اما تنها مقدار اندکی غذا در این فضا وجود دارد. هیچ‌یک از اعضای گروه از جای دقیق غذا خبر ندارند، ولی همه‌ی آنها از فاصله‌ی بین خود و غذا آگاهند. در این راستا، ساده‌ترین راه برای دستیابی به غذا دنبال‌کردن پرنده‌ای است که به غذا نزدیک‌تر است[۱۳].

هر مجموعه راه‌حل در الگوریتم، جواب‌های ممکن را جست‌وجو می‌کند. هر راه‌حل همچون پرنده‌ای وضعیت پرواز خود را براساس تجربه‌ی پرواز خود و همراهانش تنظیم می‌کند. به پاسخ به‌دست‌آمده از تجربه‌ی هر فرد به‌تنهایی، بهترین جواب فردی و به جواب به‌دست‌آمده از تجربه‌ی گروه، بهترین جواب گروهی می‌گویند. گروه پیوسته وضعیت خود را با استفاده از جواب‌های یادشده به‌روز می‌کند، بنابراین ایجاد اجتماع جدید با حرکت هرچه بیشتر به‌سمت جواب بهتر ممکن می‌شود تا اینکه سرانجام به سمت جواب بهینه همگرا شود. در عمل، تابع برازشی که از طریق

مسئله‌ی بهینه‌سازی تعیین می‌شود، میزان مطلوبیت (خوبی یا بدی) راه‌حل را مشخص می‌کند[۱۴]. نگاره‌ی ۱ فرایند به‌روزرسانی و حرکت راه‌حل‌ها را به‌صورت ترسیمی نشان می‌دهد.



نگاره‌ی ۱. فرایند به‌روزرسانی راه‌حل‌های PSO در فضای دوبُعدی

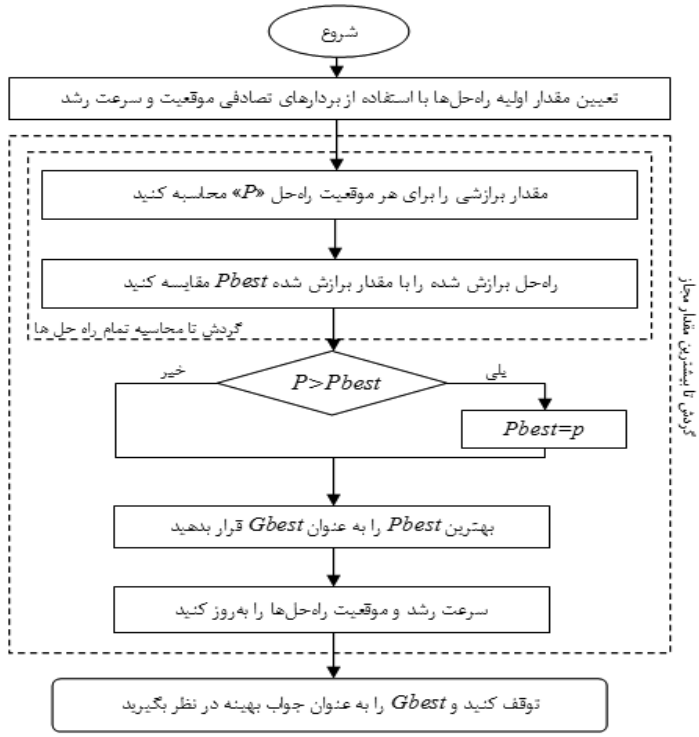
اجرای این الگوریتم از لحاظ محاسباتی در مقایسه با دیگر الگوریتم‌های ریاضی و تکاملی مؤثرتر و ساده‌تر است. این الگوریتم از ویژگی‌هایی چون همگرایی سریع، توانایی جست‌وجوی عمومی بیشتر در ابتدای پیمایش و جست‌وجوی موضعی نزدیک به پایان پیمایش برخوردار است[۲۹].

روش کار بهینه‌سازی جمعی راه‌حل‌ها به این شکل است که این الگوریتم با تعیین مقدار اولیه‌ی مجموعه‌ای از راه‌حل‌ها آغاز می‌شود. هریک از این راه‌حل‌ها با یک موقعیت $x=(x_1, x_2, \ldots, x_D)$ و یک سرعت رشد، $v=(v_1, v_2, \ldots, v_D)$، در ارتباط است. سرعت‌های رشد طبق رفتار تاریخی هر راه‌حل و همسایگانش و در زمان حرکت در فضای جست‌وجو تعدیل می‌شوند. موقعیت‌ها نیز طبق موقعیت‌های فعلی و سرعت‌های رشد در مرحله‌ی بعدی به‌روز می‌شوند. بنابراین، راه‌حل‌ها در طول فرایند

جست‌وجو به حرکت به‌سمت فضای جست‌وجوی بهتر و بهتر تمایل دارند. به‌عبارت دیگر، توصیف خلاصه‌ای از چگونگی عملکرد الگوریتم بدین‌صورت خواهد بود:

ابتدا یک راه‌حل در مقایسه با راه‌حل‌های مجاور و بر پایه‌ی تابع برازش به‌عنوان بهترین راه‌حل تعریف می‌شود، سپس همه‌ی راه‌حل‌ها در جهت این راه‌حل و همچنین به‌منظور بهترین جوابی که قبلاً پیدا کرده‌اند، به‌سرعت حرکت می‌کنند. گاهی جواب برخی از راه‌حل‌ها از حد هدف خارج می‌شود و فضای جست‌وجو را ورای بهترین راه‌حل فعلی جست‌وجو می‌کنند. در مسیر، همه‌ی راه‌حل‌ها این شانس را دارند تا به بهترین راه‌حل دست یابند که در این صورت جهت دیگر راه‌حل‌ها را تغییر داده و به‌سمت بهترین راه‌حل جدید هدایت می‌کنند. از آنجایی که بیشتر توابع از نوعی پیوستگی برخوردارند، تغییرات به جوابی بهتر منجر می‌شود. مفهوم پایه‌ای الگوریتم بر این اصل استوار است که حرکت هر راه‌حل باید به‌سمت بهترین جواب فردی[1] و بهترین جواب گروهی[2] تسریع شود. در الگوریتم بهینه‌سازی جمعی راه‌حل‌ها، هر راه‌حل، موقعیت، سرعت رشد و مقدار برازش‌شده خاص خود را دارد[۱۳]. درواقع فرمول سرعت رشد و به‌روز رسانی مکان راه‌حل‌ها هسته‌ی عملیاتی الگوریتم است[۱۶]. فلوچارت (چهارچوب) الگوریتم در نگاره‌ی ۲ نشان داده شده است.

1. Personal best (*Pbest*).
2. Global best (*Gbest*).



نگاره‌ی۲. نمودار عملکرد الگوریتم بهینه‌سازی جمعی راه‌حل‌ها

فرض کنید یک مجموعه راه‌حل شامل N راه‌حل است و $x=(x_{i,1}, x_{i,2}, \ldots, x_{i,n})$ و $v=(v_{i,1}, v_{i,2}, \ldots, v_{i,n})$ به‌ترتیب مکان و سرعت راه‌حل i که i=۱، ۲، . . . ،N است را در فضای nبُعدی، j=۱، ۲، . . . ،n، مشخص می‌کنند $Gbest= (Gbest_1, Gbest_2, \ldots, Gbest_n)$ بهترین جواب گروهی را نشان می دهند، در این صورت رابطه‌ی به‌روزرسانی سرعت هر راه‌حل به‌صورت زیر خواهد بود:

$$v_{ij}(T+1) = w.v_{ij}(T) + c_1.r_1(T).\left(Pbest_{ij}(T) - x_{ij}(T)\right) + c_2.r_2(T).\left(Gbest_j(T) - x_{ij}(T)\right)$$

که در آن T نسل تکاملی(تکرار) است، c_1وc_2 مقادیر ثابت مثبتی هستند که به‌ترتیب عامل (پارامتر) شناختی و عامل (پارامتر) اجتماعی نامیده می‌شوند، w وزن اینرسی

است و r_1 و r_2 دو عدد تصادفی بین «۱ و ۰» هستند. اساساً تساوی یادشده از سه بخش تشکیل شده است؛ بخش نخست تشکیل‌دهنده‌ی سرعت گروه است و وضعیت فعلی گروه را نشان می‌دهد؛ بخش دوم وجه شناختی است که طرز فکر فرد را در گروه بیان می‌کند و بخش سوم وجه اجتماعی است که بیانگر رفتار گروه می‌باشد[۱۴].

از وزن اینرسی برای کنترل اثر سرعت‌های رشد قبلی بر سرعت‌های رشد فعلی استفاده می‌شود. از این‌رو وزن اینرسی در تعامل بین توانایی‌های پویش موضعی و کلی راه‌حل‌ها مؤثر است[۳۰].

برای اینکه الگوریتم در ناحیه‌ی جست‌وجوی متداول حرکت کند، وزن اینرسی بزرگ‌تر، جست‌وجوی عمومی (گروهی) را میسر می‌کند و مقدار کمتر آن جست‌وجوی موضعی را تسهیل می‌کند[۳۰]، بنابراین وزن اینرسی برای تعیین نوع رفتار همگرایی الگوریتم ضروری است. معمولاً مقدار مناسب وزن اینرسی بین توانایی جست‌وجوی عمومی و موضعی توازن برقرار می‌کند و در نتیجه جواب بهینه‌ی بهتری به‌دست می‌آید[۳۱].

مطالعات بسیاری درباره‌ی بهبود میزان همگرایی الگوریتم‌های بهینه‌سازی جمعی راه‌حل‌ها[۱] صورت گرفته است. برخی از این مطالعات درباره‌ی تعیین وزن اینرسی بوده‌است. ابرهارت و کندی وزن اینرسی را در الگوریتم اولیه یک قرار دادند. «شی»[۲] به‌همراه ابرهارت و «آلاتاس»[۳] و همکارانش وزن اینرسی را [۱/۲ ، ۰/۸] در نظر گرفتند. در پژوهشی دیگر شی و همکارانش دریافتند که اگر حداکثر سرعت رشد ۲ و وزن اینرسی در دامنه‌ی «۱/۲ ، ۰/۹» باشد، به‌طور متوسط الگوریتم عملکرد بهتری از خود نشان می‌دهد[۳۲].

---

۱. از این پس به‌جای عبارت «بهینه‌سازی جمعی راه‌حل‌ها» از مخفف معادل لاتین آن یعنی PSO استفاده می‌شود.

2. Shi.

3. Alatas.

به‌تازگی طی پژوهشی وزن اینرسی به‌شکلی در نظر گرفته شد که در طول پیمایش مقدار آن به‌صورت خطی و پیوسته کاهش می‌یابد[۳۰و۳۳]. معمولاً مقدار ابتدایی وزن اینرسی ۰/۹ و مقدار نهایی آن ۰/۴ تعیین می‌شود. با تعیین «وزن اینرسی به‌صورت خطی نزولی»[1] با انجام هر تکرار، بهبود قابل ملاحظه‌ای در عملکرد PSO به‌دست آمد. «زی‌هِنگ»[2] و همکارانش [۳۴و۳۵] طی مطالعاتی اثرات به‌کارگیری نوعی تابع وزن اینرسی را بررسی کردند که با گذشت زمان وزن اینرسی افزایش یا کاهش می‌یابد.

در پژوهش‌های آنان در برخی موارد تابع صعودی وزن اینرسی در مقایسه با تابع نزولی آن نتایج بهتری از خود نشان داد. «جی‌یاؤو»[3] و همکارانش نوعی الگوریتم PSO را پیشنهاد دادند که در آن از وزن اینرسی پویا استفاده می‌شود[۳۶]. مقدار وزن اینرسی در این الگوریتم با افزایش تعداد تکرارها کاهش می‌یابد.

برخی از پژوهشگران نیز مقدار وزن اینرسی را از توزیع یکنواخت و به‌صورت تصادفی انتخاب کرده‌اند. برای مثال «زی‌هانگ»[4] و همکارانش[۳۷] مقدار وزن اینرسی را از بین اعداد تصادفی انتخاب کردند که این اعداد در بازه‌ی «۰,۱» به‌صورت یکنواختی توزیع شده بودند. این در حالی ست که ابرهارت و شی[۳۸] دامنه‌ی توزیع یکنواخت اعداد تصادفی را بین [۰/۵,۱] در نظر گرفتند.

به‌تازگی «پارک»[5] و همکارانش[۳۹] و «جی‌یانگ»[6] و «اتوره»[7][۴۰] از تابع لجستیک برای تعیین مقدار وزن اینرسی استفاده کردند. در نتیجه، وزن اینرسی طبق معادله‌ی زیر در طول فرایند تکرار تعدیل می‌شود.

1. Linear DecreasingInertia Weight (LDIW).
2. Zheng.
3. Jiao.
4. Zhang.
5. Park.
6. Jiang.
7. ETorre.

$$w(T+1) = 4w(T)(1-w(T)) \quad w(T) \in (0,1)$$

همچنین «زی‌هوو»[1] و همکارانش[۱۳] برای ایجاد بهبود در عملکرد PSO وزن اینرسی را به‌وسیله‌ی معادله‌ی زیر تعدیل کردند.

$$w = w_1 - \frac{(w_1 - w_2)}{max_{iter}} \times iter$$

از سوی دیگر پژوهش‌ها به‌سمت تعیین و تعدیل ضرایب عوامل یادگیرنده -ضرایب شتاب- معطوف شده است. «رایانویرا»[2] و «هالگاموگ»[3] [۴۱] نوعی ضریب یادگیری را معرفی کردند که در آن با گذشت زمان مقدار مؤلفه‌ی شناختی (c_1) کاهش و مقدار مؤلفه‌ی اجتماعی (c_2) افزایش می‌یابد. اگر مقدار c_1 بزرگ و مقدار c_2 کوچک در نظر گرفته شود، در ابتدای پیمایش به راه‌حل‌ها این امکان داده می‌شود تا به‌جای حرکت به‌سمت بهترین جواب فردی در فضای جست‌وجوی گسترده‌تری حرکت کنند. از طرفی در صورتی که مقدار c_1 کوچک و c_2 بزرگ در نظر گرفته شود، راه‌حل‌ها می‌توانند در مراحل پایانی جست‌وجوی PSO به مقدار بهینه‌ی عمومی همگرا شوند. معادلات ضرایب یادگیری پژوهش رایانویرا و هالگاموگ در ادامه بیان می‌شود.

$$c_1 = (c_{1i} - c_{1f}) \times \left(\frac{max_{iter} - iter}{max_{iter}}\right) + c_{1f}$$

$$c_2 = (c_{2i} - c_{2f}) \times \left(\frac{max_{iter} - iter}{max_{iter}}\right) + c_{2f}$$

به‌ترتیب c_{1i} و c_{2i} مقادیر اولیه، و c_{1f} و c_{2f} مقادیر نهایی ضریب یادگیری c_1 و c_2 است. نتایج شبیه‌سازی‌هایی در مسائل مختلف برای یافتن بهترین دامنه‌ی مقادیر c_1 و c_2 انجام شده است، این نتایج گویای آن است که بهترین جواب‌ها زمانی به‌دست می‌آیند که در طول کل دامنه‌ی جست‌وجو c_1 از ۲/۵ به ۱/۵ و c_2 از ۱/۵ به ۲/۵ تغییر کند.

1. Zhu.
2. Rayanweera.
3. Halgamug.

به‌طور کلی بهتر است c_1 و c_2 بین «۲، ۰» قرار گیرند. این پارامترها فاصله‌ای را که یک راه‌حل در یک تکرار طی خواهد کرد کنترل می‌کنند. در [۱۰] به‌طور قراردادی $c_1=c_2=2$ پیشنهاد شده است. به‌تازگی طی پژوهشی پیشنهاد شد که انتخاب پارامتر شناختی (c_1) بزرگ‌تر، بهتر خواهد بود، ولی باید $c_1+c_2=4$ باشد[۴۲]. یکی دیگر از حوزه‌هایی که پژوهشگران به آن توجه کرده‌اند، مقادیر r_1 و r_2 هستند. «کوِل‌هو»[1] و «ماریانی»[2] از تابع «هنون»[3] برای تعیین این پارامترها در الگوریتم PSO استفاده کردند[۴۳]. اخیراً از تابع لجستیک نیز برای محاسبه‌ی مقادیر r_1 و r_2 استفاده شده است[۴۴]. پس از به‌روزرسانی سرعت رشد راه‌حل، از رابطه‌ی زیر برای به‌روزرسانی مکان هر راه‌حل استفاده می‌شود:

$$x_{ij}(T+1) = x_{ij}(T) + v_{ij}(T+1) \quad i = 1, 2, \dots, N \quad j = 1, 2, \dots, n$$

مکان جدید هر راه‌حل با مقدار مکان بهینه‌ی موضعی آن پس از اتمامِ محاسباتِ مکانِ جدیدِ تمام راه‌حل‌ها مقایسه می‌شود. اگر مکان جدید از مکان بهینه‌ی موضعی بهتر باشد آنگاه بهترین جواب فردی به مکان جدید به‌روزرسانی می‌شود و در غیر این صورت مقدار فعلی آن بدون تغییر باقی می‌ماند. به‌روزرسانی بهترین جواب گروهی بر اساس جواب بهینه‌ی مجموعه راه‌حل‌ها صورت می‌گیرد و تکرار بعدی ادامه می‌یابد. پس از تعداد معینی محاسبات تکاملی به‌وسیله‌ی PSO، مجموعه راه‌حل‌ها به یک جواب بهینه همگرا می‌شوند و بنابراین یک مقدار بهینه‌ی عمومی به‌دست می‌آید.

1. Coelho.
2. Mariani.
3. Henon.

جدول۱- علایم و نشانه‌های به‌کاررفته در مدل

M	حداکثر تعداد گروه‌های طبقه‌بندی	r_1 و r_2	متغیرهای تصادفی با مقدار [۱،۰]
$\mathbf{w_k}$	وزن kاُمین مشخصه‌ی قلم i	v_{max}	حداکثر سرعت رشد راه‌حل؛ مقدار ثابت
$\mathbf{x_{(j-1,j)}}$	نقطه‌ی شکست بین گروه j-1 و طبقه گروه j که در آن $x_{(1,2)} \le x_{(2,3)} \le \dots \le x_{(M-2,M-1)} \le x_{(M-1,M)}$	N	تعداد تأمین‌کنندگان
طبقه بندی **(p,i)**	طبقه‌بندی گروهی که در آن قلمi در راه حلP قرار دارد	N	تعداد اقلام
T	شماره‌ی تکرار	A_l	هزینه‌ی سفارش اصلی برای سفارش تأمین‌شده به‌وسیله‌ی lاُمین تأمین‌کننده
$\mathbf{ws_{(p,i)}}$	امتیاز وزنی قلم i در راه‌حل P	a_i	هزینه‌ی راه‌اندازی قلم i
$\mathbf{v_{(j-1,j)}^{(T+1)}}$	سرعت رشد نقطه‌ی شکست بین گروه j-1 و طبقه‌ی گروه j در T+1اُمین تکرار (بعد از به‌روز رسانی)	C_j	مجموعه‌ی تأمین‌کنندگانی که اقلام طبقه‌ی j را تدارک می‌بینند
$\mathbf{v_{(j-1,j)}^{(T)}}$	سرعت رشد نقطه‌ی شکست بین گروه j-1 و طبقه‌ی گروه j در Tاُمین تکرار (قبل از به روز رسانی)	T_j	مدت زمان سفارش طبقه‌ی j
$\mathbf{x_{(j-1,j)}^{(T+1)}}$	موقعیت نقطه‌ی شکست بین گروه j-1 و طبقه‌ی گروه jدر T+1اُمین تکرار(بعد از به روزرسانی)	D_i	تقاضا بر حسب واحد زمانی برای قلم i
$\mathbf{x_{(j-1,j)}^{(T)}}$	موقعیت نقطه‌ی شکست بین گروه j-1 و طبقه‌ی گروه j در Tاُمین تکرار(قبل از به روزرسانی)	h_i	هزینه‌ی نگهداری هر واحد موجودی بر حسب زمان برای قلم i
W	وزن اینرسی	S_j	مجموعه‌ی اقلام متعلق به طبقه‌ی j
Pbest	بهترین مقدار برازش‌شده‌ی هر راه‌حل - بهترین جواب فردی	TRC	هزینه‌ی کل مرتبط
Gbest	بهترین مقدار برازش‌شده‌ی عمومی - بهترین جواب گروهی	ITR	نرخ گردش موجودی
$\mathbf{c_1}$ و $\mathbf{c_2}$	عوامل یادگیرنده؛ c_1 پارامتر شناختی و c_2 پارامتر اجتماعی	MO	تابع چندهدفه

تابع هزینه[1]

ازجمله هزینه‌های مرتبط با موجودی که در این مقاله بدان‌ها توجه شده است و از اهمیت بسیاری در مدیریت موجودی برخوردارند، می‌توان به هزینه‌ی سفارش‌دهی اصلی برای هر سفارشی که از سوی تأمین‌کننده‌ی مشخص فراهم می‌شود، هزینه‌ی راه‌اندازی هر قلم موجودی در زمان پرکردن دوباره‌ی انبار و هزینه‌ی نگهداری اقلام در انبار اشاره کرد. فرض شده است که اقلام طبقه‌بندی‌شده در هر گروه مشابه، چرخه‌ی بازپُرسازی مشابهی دارند؛ یعنی یک سفارش مشترک برای برطرف‌کردن نیازمندی‌های اقلام هر گروه انجام می‌شود.

«چاکراوارتی»[2] [۴۵] نشان داد که هزینه‌های راه‌اندازی چرخه‌ی بازپُرسازی انبار از اقلام گروه j برابر با $\sum_{i\in S_j} a_i$ است. وی همچنین نشان داد که چرخه‌ی مشترک و بهینه‌ی بازپرسازی انبار از اقلام گروه jبه‌صورت معادله‌ی $T_j = \sqrt{\frac{2\sum_{i\in S_j} a_i}{\sum_{i\in S_j} D_i h_i}}$ قابل بیان است و هزینه‌های کل مرتبط با اقلام گروه j به‌صورت معادله‌ی $TRC_j = \frac{\sum_{i\in S_j} a_i}{T_j} + \frac{1}{2} T_j \sum_{i\in S_j} D_i h_i$ نوشته می‌شود.

لازم به یادآوری است که هزینه‌های سفارش‌دهی اصلی در مطالعه‌ی وی مد نظر قرار نگرفت، ولی در این پژوهش فرض شده است که هنگام ایجاد یک سفارش مشترک برای بازپرسازی انبار از اقلام یک گروه، هزینه‌های سفارش‌دهی اصلی در خصوص تأمین‌کنندگان درگیر در سفارش روی می‌دهند. به‌علاوه، هر قلم موجود در این سفارش مشترک، یک هزینه‌ی راه‌اندازی ایجاد می‌کند. بنابراین، مجموع هزینه‌های راه‌اندازی و سفارش‌دهی اصلی در هر چرخه‌ی بازپرسازی انبار از گروه j برابر با $\sum_{i\in S_j} a_i + \sum_{l\in C_j} A_l$ هستند. از این‌رو، چرخه‌ی مشترک و بهینه‌ی بازپرسازی انبار

1. Cost Function.
2.Chakravarty.

از اقلام گروه j به‌صورت $T_j = \sqrt{\frac{2(\sum_{i\in S_j} a_i + \sum_{l\in C_j} A_l)}{\sum_{i\in S_j} D_i h_i}}$ قابل تجدید نظر است. سرانجام هزینه‌های کل مرتبط با سیستم به‌صورت $TRC_j = \frac{\sum_{i\in S_j} a_i + \sum_{l\in C_j} A_l}{T_j} + \frac{1}{2} T_j \sum_{i\in S_j} D_i h_i$ بیان می‌شود.

نسبت گردش موجودی[1]

نسبت گردش موجودی معیاری رایج برای اندازه‌گیری عملکرد در مدیریت موجودی است. نسبت‌های بیشتر نشان می‌دهند که موجودی برای مدت زمان طولانی در انبار نمانده، بلکه به‌سرعت پس از دریافت استفاده شده‌اند. بنابراین، هدف به‌حداکثررساندن این نسبت است. میزان بالای گردش موجودی حاکی از نرخ بالای بازگشت سرمایه‌گذاری در موجودی است. بااین‌حال، اگرچه گردش موجودی، سطوح مختلف موجودی را به فعالیت‌های فروش مرتبط می‌کند، ولی منافع به‌دست‌آمده از موجودی را منعکس نمی‌کند. این نسبت به‌صورت زیر تعریف می‌شود:

$$\text{نسبت گردش موجودی} = \frac{\text{فروش یا بهای تمام شده سالیانه}}{\text{متوسط موجودی}}$$

متوسط موجودی قلم i در گروه j برحسب واحد زمان به‌صورت $\frac{D_j T_j}{2}$ بیان می‌شود، بنابراین مجموع متوسط موجودی همه‌ی اقلام، $\sum_{j=1}^{M} \sum_{i\in S_j} \frac{D_i T_j}{2}$ خواهد بود. سپس معادله‌ی نسبت گردش موجودی با استفاده از موارد یادشده بدین‌صورت بیان می‌شود:

$$\text{نسبت گردش موجودی} = \frac{\sum_{i=1}^{n} D_i}{\sum_{j=1}^{M} \sum_{i\in S_j} (D_i h_i / 2)} = \frac{2\sum_{i=1}^{n} D_i}{\sum_{j=1}^{M} \sum_{i\in S_j} D_i h_i}$$

1. Inventory Turnover Ratio (ITR).

تابع چندهدفه

هر تابع تک‌هدفه، به‌دنبال هدفی مشخص است و به نیازمندی‌های خاصی پاسخ می‌دهد، اما در مدیریت موجودی باید به عوامل متعددی توجه داشت و بنابراین تابع تک‌هدفه برای ایجاد نتایج رضایت‌بخش طبقه‌بندی موجودی با توجه به معیارهای گوناگون، نامناسب است. از این‌رو باید توابع چندهدفه را در الگوریتم دخالت داد. روش‌های بسیاری را می‌توان برای تعیین توابع چندهدفه به‌کار گرفت. در این مقاله از روش برنامه‌ریزی هدف[1] استفاده شده است.این روش به‌وسیله‌ی «چارنس»[2] و «کوپر»[3] در سال ۱۹۹۱ پیشنهاد شد. برای ترکیب دو هدف یادشده (تابع هزینه و نرخ گردش موجودی)، در ابتدا مقدار هر تابعِ هدف به‌صورت زیر، نرمال‌سازی می‌شود تا مقدار آن بین ۰ و ۱ قرار گیرد.

$$\frac{\text{مقدار تابع هدف پس از طبقه‌بندی}}{\text{مقدار تابع هدف اگر همه‌ی اقلام در یک گروه قرار گیرند}}$$

روش برنامه‌ریزی هدف به‌دنبال حداقل‌سازی تفاوت بین مقادیر هدف و مقادیر ثابت است. اگر مسئله را به‌صورت تک‌هدفه و بدون توجه به دیگر اهداف حل کنیم، بهترین مقدار هر تابع تک‌هدفه به‌عنوان مقدار ثابت آن تابع در تابع چندهدفه در نظر گرفته می‌شود. سپس سعی می‌شود تا فاصله‌ی بین مقدار توابع، هنگامی که هم‌زمان مد نظر قرار می‌گیرند با زمانی که به‌تنهایی محاسبه می‌شوند (مقادیر ثابت) حداقل شود. هدف در این مسئله‌ی چندهدفه حداقل‌کردن مقادیر زیر است:

1-Goal Programming
2-Charnes

$$\text{هدف} = \left(\text{مقدار ثابت تابع هزینه(تک‌هدفه)} - \text{مقدار هدف تابع هزینه(چندهدفه)}\right) + \left(\frac{1}{\text{مقدار ثابت تابع گردش موجودی (تک‌هدفه)}} - \frac{1}{\text{مقدار هدف تابع گردش موجودی (چندهدفه)}}\right)$$

از آنجایی که هدف حداکثرکردن گردش موجودی است، مقادیر معکوس این هدف استفاده شده است، زیرا تابع چندهدفه از نوع حداقل‌سازی است.

متدولوژی پژوهش

الگوریتم با ایجاد مجموعه راه‌حل‌های اولیه شروع می‌شود. در هر تکرار مقدار مناسب هر راه‌حل طبق تابع هدف طراحی شده و موقعیت فعلی آن ارزیابی می‌شود، سپس سرعت رشد آن محاسبه شده و حرکت بعدی راه‌حل را در فضای جست‌وجو تعیین می‌کند. محاسبه براساس بهترین جواب فردی و بهترین جواب گروهی که تاکنون وجود داشته انجام می‌شود. موقعیت جدید هر راه‌حل به‌وسیله‌ی آخرین سرعت رشدش تعیین می‌شود. تکرارها تا زمان دستیابی به معیار توقف ادامه می‌یابد. هنگام اجرای طبقه‌بندی اقلام، این الگوریتم ابتدا همه‌ی اقلام را طبق امتیاز وزنی‌شان دسته بندی کرده و پس از آن نقاط شکست در امتداد فهرست گروه‌بندی تعیین می‌شوند. اقلام بین دو نقطه‌ی شکست مجاور در گروه‌های مشابهی قرار می‌گیرند.

مشخصات هر قلم از قبیل نرخ تقاضا و هزینه‌ی نگهداری هر واحد هنگام تعیین نقاط شکست مناسب مد نظر قرار می‌گیرند. به هر مشخصه برای ترکیب‌شدن با اهداف مختلف طبقه‌بندی، وزنی اختصاص داده می‌شود تا درجه‌ی ارتباط آنها با اهداف منعکس شود. همچنین این اوزان به‌عنوان بخشی از فضای جست‌وجو طراحی می‌شوند تا اوزان مناسب تعیین شود. از این‌رو فضای جست‌وجو $P = (w_1, w_2, w_3, \dots , w_k, x_{(1,2)}, x_{(2,3)}, \dots , x_{(M-1, M)})$ خواهد بود که در آن $x_{(j-1,j)}$ نقطه‌ی شکست بین طبقه‌ی j-1 و j و w_k وزن اختصاص‌داده‌شده به مشخصه‌های اقلام است. مقدار هر موقعیت و سرعت رشد ابتدایی برای بخش اوزان فضای جست‌وجو به‌صورت تصادفی و

بین «۰.۱» ایجاد می‌شود. مقدار ابتدایی موقعیت نقطه‌ی شکست باید به‌صورت $x_{(1,2)}$ $\leq x_{(2,3)} \leq \ldots \leq x_{(M-1, M)}$ باشد و به‌صورت تصادفی و بین [۰،۱] ایجاد شود. امتیاز وزنی اقلام از طریق دنبال‌کردن قانونی ساده که به‌وسیله‌ی گوونیر و ارل[۶] پیشنهاد شده، محاسبه می‌شوند. در ابتدا مقادیر مشخصات هر قلم به‌صورت $\frac{i_k-min_k}{max_k-min_k}$ نرمال سازی می‌شوند تا اعداد بین «۰،۱» به‌دست آید. i_k مقدار ارزش kاُمین مشخصه‌ی قلم i است (برای مثال هزینه‌ی نگهداری). min_k و max_k مقادیر حداقل و حداکثر ارزش مشخصه‌یk در میان همه‌ی اقلام هستند. سپس امتیاز وزنی قلم i در راه‌حل P از رابطه‌ی $ws(p,i) = \sum_{k=1}^{k} w_k \frac{i_k-min_k}{max_k-min_k}$ محاسبه می‌شود. با مقایسه‌ی امتیاز وزنی هر قلم با نقاط شکست، گروه‌بندی اقلام صورت می‌گیرد؛ یعنی:

$$\text{بندی‌طبقه}(p,i)\begin{cases}1, \text{if } 0/00 < ws(p,i) \leq x_{(1,2)}\\ 2, \text{if} x_{(1,2)} < ws(p,i) \leq x_{(2,3)}\\ 3, \text{if} x_{(2,3)} < ws(p,i) \leq x_{(3,4)}\\ \vdots\\ M, \text{if} x_{(M-1,M)} < ws(p,i) \leq 1\end{cases}$$

نتیجه‌ی طبقه‌بندی که به‌صورت یک راه‌حل نمایش داده می‌شود، از طریق مقادیر اوزان (w_k) و نقاط شکست در آن راه‌حل تعیین می‌شود. مقدار مناسب راه‌حل یادشده، مقدار تابع هدف متناظر با نتیجه‌ی آن طبقه‌بندی است. پس از محاسبه‌ی مقدار مناسب هر راه‌حل، بهترین جواب فردی و گروهی به‌روز می‌شوند. برای به‌روزرسانی سرعت رشد و موقعیت هر راه‌حل به ترتیب از معادلات زیر استفاده می‌شود:

$$v_{(j-1,j)}^{(T+1)} = w.v_{(j-1,j)}^{(T)} + c_1 r_1 \left(Pbest - x_{(j-1,j)}^{(T)}\right) + c_2 r_2 \left(Gbest - x_{(j-1,j)}^{(T)}\right)$$

اگر $v_{max} < v_{(j-1,j)}^{new}$ آنگاه $v_{(j-1,j)}^{new} = v_{max}$ و اگر $-v_{max} > v_{(j-1,j)}^{new}$ آنگاه $v_{(j-1,j)}^{new} = -v_{max}$.

هنگام به‌روزرسانی، محدودیت $x_{(1,2)} \leq x_{(2,3)} \leq \ldots \leq x_{(M-1,\ M)}$ نباید نقض شود، بنابراین اصلاحیه‌ی زیر اعمال می‌شود.

اگر $x_{(j-2,j-1)} > x_{(j-1,j)}$ آنگاه $x_{(j-2,j-1)} = x_{(j-1,j)}$ و $3 \leq j \leq M$.

در ادامه گام‌های الگوریتم بیان شده است:

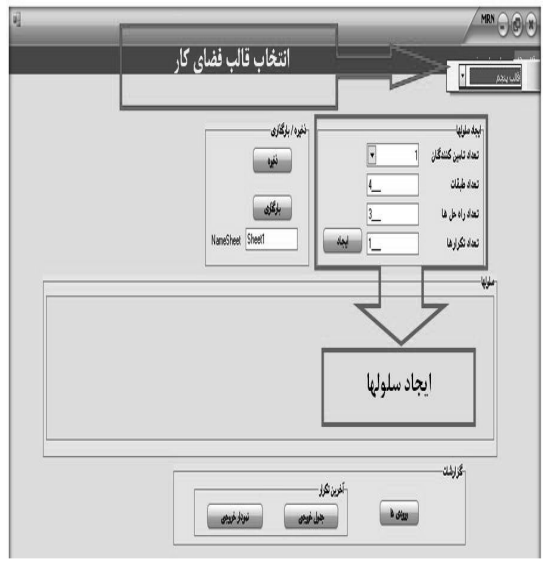
گام ۱. فراهم‌کردن مقدمات شروع الگوریتم؛ گام ۲. محاسبه‌ی امتیازات وزنی؛ گام ۳. طبقه‌بندی اقلام؛ گام ۴. محاسبه‌ی مقادیر تابع برازش؛ گام ۵. به‌روزرسانی بهترین جواب فردی و بهترین جواب گروهی؛ گام ۶. به‌روزرسانی سرعت رشد و موقعیت؛ گام ۷. اگر به معیار پایانی دست یافته‌اید توقف کنید، در غیر این صورت به گام ۲ برگردید.

همیشه تعداد گروه‌های به‌دست‌آمده به‌وسیله‌ی الگوریتم پیشنهادی، کوچک‌تر یا مساوی با حداکثر تعداد گروه‌های طبقه‌بندی مجموعه‌ی ابتدایی است، بنابراین، بهتر است برای الگوریتم تعداد گروه حداکثری انتخاب شود تا این اطمینان به‌دست آید که تعداد گروه بهینه قابل دستیابی است. با افزایش تعداد گروه‌های اقلام موجودی، میزان پیچیدگی مدیریت موجودی نیز افزایش می‌یابد. در نتیجه، معمولاً مدیران مایل هستند گروه‌های طبقه‌بندی کوچک‌تری داشته باشند. از این‌رو ممکن است برخی در ابتدا بزرگ‌ترین تعداد گروه قابل قبول را تعیین کرده و سپس آن را به‌عنوان تعداد حداکثر گروه طبقه‌بندی مد نظر قرار دهند. تعداد گروه به‌دست‌آمده از الگوریتم، کوچک‌تر یا مساوی با بزرگ‌ترین تعداد گروه‌های قابل قبول خواهد بود. این تعداد گروه، نه‌تنها به اهداف تعیین‌شده دست می‌یابد، بلکه در محدوده‌ی پذیرش نیز قرار می‌گیرد.

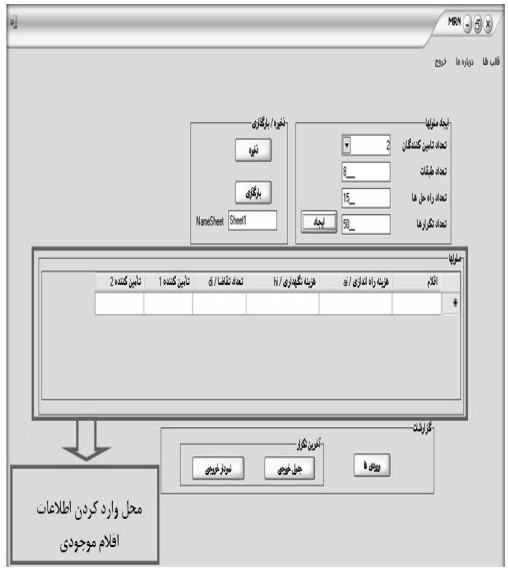
معرفی نرم‌افزار

همان‌طور که در نگاره‌ی ۳ می‌بینید زیر بخش «ایجاد سلول‌ها» کاربر می‌بایست

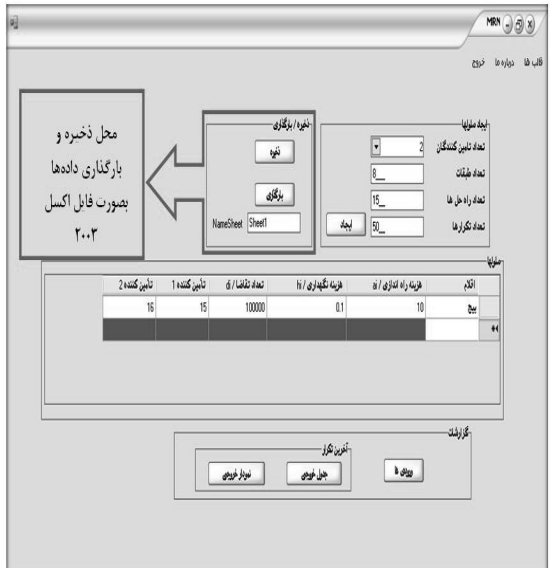
تعداد تأمین‌کنندگان، طبقات، راه‌حل‌ها و تکرارها را تعیین کند. پس از انجام این کار روی «ایجاد» کلیک کرده و در ادامه داده‌های مربوط به اقلام موجودی را در بخش «سلول‌ها» وارد کرده یا داده‌های ثبت‌شده در اکسل را به برنامه اضافه می‌کند (نگاره‌ی ۴ و ۵). رمز ورود به برنامه، نام آن یعنی MOPSO است.



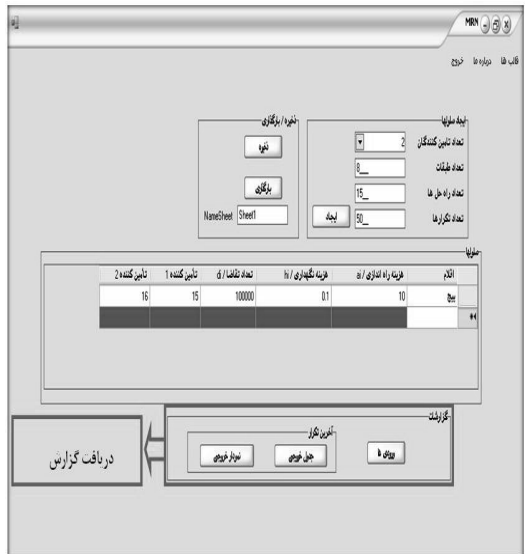
نگاره‌ی ۳. انتخاب قالب فضای کار و ایجاد سلول‌ها



نگاره‌ی ۴. ایجاد سلول به منظور واردکردن داده‌ها



نگاره‌ی ۵. واردکردن داده‌ها و ذخیره‌ی آنها



نگاره‌ی ۶. دریافت گزارش

در انتها برای دریافت خروجی و نتایج عملیات الگوریتم کافی است زیر بخش «گزارشات» حسب اطلاعات مورد نظر، بر روی گزینه‌ی «ورودی‌ها» برای چاپ داده ها، گزینه‌ی «جدول خروجی» برای چاپ نتایج اجرای الگوریتم و گزینه‌ی «نمودار خروجی» برای مقایسه‌ی جواب راه‌حل‌های مختلف کلیک کند(نگاره‌ی ۶). کار اصلی نرم‌افزار این است که با تجزیه‌وتحلیل ورودی‌ها، تمامی راه‌حل‌های ممکن با هدف‌های مشابه (تعداد طبقات، نرخ گردش موجودی‌ها و مجموع هزینه‌ها) را به‌عنوان خروجی به‌دست می‌دهد که تصمیم‌گیرنده با بررسی این راه‌حل‌ها و اهداف چندگانه به‌راحتی می‌تواند بهترین آنها را تشخیص دهد و انتخاب کند.

داده‌های الگوریتم

به‌منظور آزمایش الگوریتم پس از تدوین برنامه، مقادیر تقاضا، هزینه‌های راه‌اندازی، نگهداری و سفارش‌دهی برای ۱۰۰ قلم موجودی فرضی به‌صورت تصادفی در اکسل

ایجاد شد. دامنه‌ی تغییر هر یک از پارامترها بدین‌شرح است:
هزینه‌ی راه‌اندازی: [10,1200]، هزینه‌ی نگهداری: [0,1]، نرخ تقاضا: [50,6000] و هزینه‌ی سفارش‌دهی: [100,2000]؛ البته شایان یادآوری است که در عالم واقع اگر به موجودی‌ها دقت شود، مشاهده می‌شود که اقلام موجودی از توزیع پارتو تبعیت می‌کنند؛ یعنی، ۱۰درصد اقلام حدوداً ۶۵درصد، ۳۵درصد اقلام ۳۰درصد و ۳۵درصد دیگر اقلام فقط ۵درصد ارزش ریالی را به‌خود اختصاص می‌دهند که بر این اساس به‌ترتیب به سه طبقه‌ی A، B و C تقسیم می‌شوند. بنابراین برای آزمایش الگوریتم در عالم واقع، می‌توان داده‌های مورد نیاز را از شرکتی واقعی تهیه کرد. هدف این مقاله صرفاً ارائه‌ی روشی نو برای طبقه‌بندی است که قادر است هم‌زمان چندین هدف را مد نظر قرار دهد. داده‌های مثال عددی مقاله در جدول ۲ آمده است.

اقلام	هزینه‌ی راه‌اندازی	هزینه‌ی نگهداری	تقاضا	تأمین‌کننده‌ی اول	تأمین‌کننده‌ی دوم
۱	1,040.496	0.336619159520249	2897.79961546678	1629.93865779595	**727.225562303537**
۲	1,191.284	0.466658528397473	2283.49711600085	1230.30487990967	**752.275154881436**
۳	560.639	0.101260414441359	4295.46037171545	701.886043885617	**1179.50987273782**
۴	548.690	0.700918607135227	4931.37150181585	376.473281044954	**1057.79900509659**
۵	1,117.125	0.158513138218329	147.692800683615	812.347788933988	**391.433454390088**
۶	856.078	0.214728232673116	4436.00726340525	1892.95937986389	**990.072328867458**
۷	676.091	0.662923062837611	3452.36060670797	1534.78499710074	**730.066835535752**
۸	541.718	0.256355479598376	4852.92672505875	863.722647785882	**148.8235114597**
۹	1,163.756	0.796136356700339	1197.25486007263	1648.14600054933	**987.868892483291**
۱۰	1,078.520	0.313760795922727	2017.11172826319	527.524643696402	**1899.97558519242**
۱۱	1,174.215	0.558580278939177	4951.52745139927	902.862636188849	**980.272835474715**
۱۲	694.794	0.30558183538316	4531.3394573809	1851.67394024476	**927.216406750694**
۱۳	561.619	0.496810815759758	5127.30185857723	1085.28397473067	**659.962767418439**
۱۴	814.640	0.49687185277871	309.666737876522	1681.9513534959	**1778.90255439924**
۱۵	201.754	0.663655507065035	245.567186498611	1836.82973723563	**1306.72933133946**
۱۶	697.300	0.497909482100894	4728.35932493057	275.347148045289	**610.617389446699**
۱۷	599.026	0.984862819299905	3960.25421918394	686.056093020417	**1762.550737022**
۱۸	486.370	0.0740379039887692	1644.49903866695	1716.51051362651	**1089.28495132298**
۱۹	888.908	0.0992767113254189	2519.37620166631	1663.28012939848	**1492.33985412152**
۲۰	1,176.757	0.816400646992401	3535.34501174963	282.131412701804	**1090.73458052309**
۲۱	874.454	0.772576067384869	566.972869045076	578.609576708274	**1549.28128910184**
۲۲	1,019.759	0.623981444746239	3001.84789574877	1487.64305551317	**233.365886410108**
۲۳	697.409	0.282448805200354	1078.3165990173	741.837824640645	**1990.02655110324**
۲۴	102.826	0.0513931699575793	68.7032685323649	556.111331522568	**779.760124515519**
۲۵	118.479	0.864253669850764	4271.491134373	670.226142155217	**721.19510483108**
۲۶	338.488	0.507522812585833	1015.48814355907	1043.94054994354	**745.722830896939**
۲۷	304.023	0.509781182287057	632.888271736808	564.055299539171	**1533.04544206061**
۲۸	796.336	0.675008392590106	5290.72847682119	260.502945036164	**116.525772881252**
۲۹	281.252	0.258400219733268	5084.08459730827	1108.13013092441	**1395.15671254616**
۳۰	291.057	0.985259559923093	1588.933988464	1588.933988464	**225.247962889492**
۳۱	223.508	0.730521561326945	5090.25849177526	771.584215826899	**226.523636585589**
۳۲	788.492	0.146458326975311	5100.42725913266	619.895016327403	**129.978331858272**
۳۳	237.018	0.155705435346538	1869.84618671224	620.64882351146	**1201.7181920835**

778.832361827448	863.43272194586	4563.11685537278	0.219458601641896	694.068	۳۴
104.34888760033	1804.18408764916	4797.3616748558	0.665395062105167	381.087	۳۵
1005.72832422864	1013.55632190924	2322.53791924802	0.181157872249519	392.600	۳۶
672.139652699362	1824.36292611469	2078.30591753899	0.277809991760002	564.561	۳۷
397.92779320658	923.447370830409	2292.2132023072	0.942258980071413	611.664	۳۸
1785.91875972777	1836.53981139561	5354.8280281991	0.927274391918699	589.039	۳۹
953.657643360698	802.374340037233	5389.32920316172	0.0132755516220588	232.623	۴۰
223.392437513352	1230.36286507767	2892.35206152532	0.203833124790185	993.719	۴۱
341.334269234291	1596.94509720145	4218.83144627216	0.0708029419843135	213.157	۴۲
616.473891415143	413.583788567766	2238.28241828669	0.300027466658528	477.291	۴۳
423.035370952483	1961.26590777306	1088.12219611194	0.184026612140263	410.904	۴۴
100.869777520066	1535.71275978881	2323.99060029908	0.130954924161504	77.005	۴۵
1744.0534684286	278.884243293558	1353.59965819269	0.0513931699575793	717.928	۴۶
1827.43614001892	1243.11960203864	4801.53813287759	0.832148197882015	402.079	۴۷
561.561937314981	938.233588671529	5279.10702841273	0.886593218787194	309.979	۴۸
488.094729453413	1132.19397564623	2512.29438154241	0.964659566026795	97.197	۴۹
1725.78814050722	1534.90096743675	1749.81841486862	0.212988677632984	531.040	۵۰
1097.75078585162	1855.26902066103	1995.68468276009	0.900601214636677	693.269	۵۱
965.718558305612	913.183996093631	2847.50053407392	0.398602252265999	1,000.293	۵۲
1842.86019470809	319.125949888607	204.165776543474	0.145512253181555	588.675	۵۳
220.493179113132	600.99185155797	3281.67058320872	0.911832026123844	547.819	۵۴
1354.45112460707	588.814966277047	3129.86541337321	0.901944029053621	764.159	۵۵
819.711905270547	576.75405133۲	1697.34031189917	0.634449293496506	918.761	۵۶
768.858912930692	1952.85805841243	4020.7220679342	0.118808557390057	144.736	۵۷
1198.29706717124	673.067415387432	5062.65755180517	0.748344370860927	70.976	۵۸
793.154698324534	4358.83358256783	4358.83358256783	0.726615192114017	204.115	۵۹
868.767357402264	102.725302896207	154.411450544755	0.347117526779992	1,070.094	۶۰
482.992034669027	417.9906613361	2090.29053621021	0.852412488174078	597.755	۶۱
501.837214270455	629.520554216132	1165.11429181799	0.637775811029389	1,078.592	۶۲
156.07165746025	544.688253425703	2317.81670583209	0.67253639332255	998.368	۶۳
824.350718710898	1334.50422681356	3329.79064302499	0.0123905148472549	896.208	۶۴
786.37043366802	1254.94857631153	2952.6383251442	0.310159611804559	1,114.837	۶۵
970.125431073946	987.347025971252	4426.74642170476	0.0151066621906186	1,096.206	۶۶
349.626148258919	368.819238868374	797.585985900449	0.954680013428144	1,094.535	۶۷
1048.57936338389	1058.37885677664	865.498825037385	0.925504318369091	1,064.828	۶۸

1799.83520004883	727.689443647572	1913.78978850673	0.286538285470138	95.345	۶۹
1469.66765343181	835.657826471755	2355.04165776543	0.956511123996704	742.369	۷۰
436.777855769524	624.939725943785	973.905148472549	0.466170232245857	1,071.147	۷۱
1213.6631366924	1885.36332285531	5913.92864772485	0.542130802331614	273.226	۷۲
1572.30140079959	1866.34418774987	1705.87481307413	0.508743552964873	743.858	۷۳
506.707968382824	1245.67094943083	5966.04358043153	0.497665334025086	743.858	۷۴
1289.33378093814	864.998321481979	4386.43452253792	0.948088015381329	1,037.191	۷۵
983.983886226997	1612.77504806665	2318.17987609485	0.594714194158757	1,128.238	۷۶
922.287667470321	1968.86196478164	5082.81350138859	0.412671285134434	123.854	۷۷
941.01687673574	1202.70393993957	4316.88741721854	0.930875576036866	321.455	۷۸
232.728049562059	144.59059419538	1332.89895321512	0.223273415326395	1,187.870	۷۹
1117.69768364513	788.457899716178	1001.32450331126	0.0154423657948546	297.703	۸۰
1320.64577166051	1496.57277138585	1104.28327280496	0.499130222479934	406.364	۸۱
1374.10809656056	1557.63115329447	3420.4016235847	0.361674855800043	664.578	۸۲
969.835505233924	1556.52943510239	3621.2347788934	0.65126499221778	995.680	۸۳
534.192938016907	749.027985473189	4927.37662892544	0.197180089724418	605.781	۸۴
1575.25864436781	1647.45017853328	5541.86071352275	0.676107058931242	699.079	۸۵
633.231604968413	1144.66078676717	3950.26703695792	0.306833094271676	851.502	۸۶
1316.29688406018	364.470351268044	5096.06921597949	0.459700308236946	942.258	۸۷
1045.21622363964	1453.43180639058	339.809869685965	0.623310037537767	36.257	۸۸
1926.35883663442	641.523483993042	4605.06302072207	0.493850520340587	1,029.092	۸۹
922.983489486374	561.851863155003	1224.49262977996	0.614215521713919	420.346	۹۰
132.645649586474	449.244666890469	282.247383037812	0.10940885647145	580.214	۹۱
1995.07126071963	1785.10696737571	5700.20294808802	0.398236030152287	76.170	۹۲
411.612292855617	1603.38145084994	5225.72099978637	0.678579058198798	666.103	۹۳
1866.92403942991	1562.09601123081	4677.69707327494	0.486373485518967	483.974	۹۴
1187.97570726646	223.392437513352	4064.12091433454	0.348551896725364	399.427	۹۵
308.514664143803	242.991424298837	4471.05319376202	0.709494308297983	265.163	۹۶
1511.47495956297	275.115207373272	5398.04528946806	0.871517075106052	938.517	۹۷
1410.11688589129	450.288399914548	5606.50502029481	0.888943144016846	191.040	۹۸
238.352610858486	326.316110721152	438.41059602649	0.112857448042238	98.069	۹۹
992.10180974761	594.903408917509	1421.51249732963	0.752494888149663	1,151.335	۱۰۰

نتایج و یافته‌ها

پس از واردکردن داده‌های تولیدشده در برنامه، حداکثر طبقات ۴ و تعداد راه‌حل‌ها و تکرارها به‌ترتیب ۱۰ و ۱۰ تعیین شد. نخستین چیزی که در جدول ۱ مشاهده می‌شود و برای ما پرسش ایجاد می‌کند مقدار صفر راه‌حل‌های نهم و دهم است. مفهوم صفر در این جدول این است که این دو راه‌حل جوابی مناسب به‌دست نیاورده اند. به‌عبارت دیگر، هنگام به‌روزرسانی نقاط شکست، به دلیل حرکت این راه‌حل‌ها در خلاف جهت و در نتیجه‌ی منفی‌شدن سرعت رشد، مقادیر نقاط شکست نیز منفی شده‌اند. به مثال پرندگان باز گردیم. فرض کنید دو فرد از گروه به‌جای حرکت همراه گروه و در جهت غذا، در جهت خلاف حرکت کنند. نتیجه‌ی این عمل چیزی جز محروم‌شدن از غذا نیست، بنابراین مقدار منفی سرعت رشد و درنتیجه‌ی آن نقطه‌ی شکست بین طبقات سبب می‌شود که هیچ طبقه‌بندی صورت نگیرد و دلیل این امر مثبت‌بودن ذاتی مقدار ws(p,i) است.

از سوی دیگر کمترین مقدار هزینه‌ی کل مرتبط با فرایند طبقه‌بندی موجودی، مقدار مربوط به راه‌حل اول و برابر 589,540,002.410 است. به این مقدار «بهترین جواب گروهی» می‌گویند؛ یعنی بهترین مقدار تمام راه‌حل‌ها در پایان تکرار فعلی. شایان ذکر است این مقدار زمانی به‌دست می‌آید که تابع هزینه به‌تنهایی در نظر گرفته شود. در ستون دوم که تعداد طبقات را نشان می‌دهد عدد ۳ بیانگر تعداد طبقات مربوط به این راه‌حل پس از اتمام ۱۰ تکرار است.

با نگاهی به ستون چهارم (ستون مربوط به بررسی تابع تک‌هدفه‌ی نرخ گردش موجودی) می‌توان دریافت که راه‌حل هشتم با طبقاتی به اندازه‌ی ۲، کمترین مقدار را در بین جواب‌های تمام راه‌حل‌ها دارد.

در ستون پنجم، مقدار تابع چندهدفه آمده که الگوریتم به‌دنبال حداقل‌کردن آن است. مشاهده می‌شود که مقدار راه‌حل هشتم کمینه بوده و بنابراین پاسخ نهایی

ماست. تعداد طبقات مربوط به این راه‌حل ۲ است. این بدان معنی است که راه‌حل هشتم با قراردادن این ۱۰۰ قلم موجودی در ۲ طبقه توانسته مقدار تابع چندهدفه را حداقل کند. این در حالی ست که در این شرایط مقدار نرخ گردش موجودی نیز حداکثر است.

جدول ۳. نتیجه‌ی محاسبات در تکرار آخر

راه‌حل	تعداد طبقات	**TRC**	**ITR**	**MO**
۱	3	589,540,002.410	3,239,362.079	1,938,743.465
۲	3	614,291,687.261	4,222,390.823	2,526,416.173
۳	3	817,692,559.223	2,436,962.609	1,460,126.498
۴	3	999,328,269.970	460,205.084	279,383.464
۵	3	1,057,074,577.901	1,107,326.247	666,423.389
۶	3	1,219,246,999.898	847,609.984	511,888.599
۷	3	1,324,428,512.812	616,126.022	373,982.483
۸	2	1,565,112,681.725	363,403.422	223,966.833
۹	0	0.000	0	0
۱۰	0	0.000	0	0

TRC: هزینه‌ی کل مرتبط
ITR: نرخ گردش موجودی
MO: تابع چندهدفه

نتیجه‌گیری و پیشنهادات

یکی از مهم‌ترین اهداف مدیریت موجودی طبقه‌بندی اقلام موجودی به‌نحوی است که مدیریت را قادر کند تا به اهدافی چون کنترل دقیق‌تر و آسان‌تر موجودی‌ها، کاهش هزینه‌ها و افزایش گردش موجودی‌ها دست یابد. از این‌رو پژوهشگران این حوزه، راه‌های متفاوتی را برای نیل به این مهم پیموده‌اند.

در همه‌ی این روش‌های ابتکاری ارائه‌شده تاکنون، پژوهشگر با بررسی‌های مقدماتی،

ابتدا شاخص‌های مهم از نظر مدیران موجودی را شناسایی کرده و پس از تعیین مقادیر شاخص‌های بااهمیت، با استفاده از روش مورد نظر آنها را برای نمونه‌ی انتخابی از موجودی، تجزیه‌وتحلیل کرده‌اند. در پایان برتری‌های روش مورد نظر خودشان را در مقایسه با روش‌های قدیمی‌تر ارائه کرده‌اند. اما در این مقاله نخست اینکه محدودیتی در تعداد طبقه‌ها وجود ندارد، زیرا روش ارائه‌شده پس از تجزیه‌وتحلیل داده‌ها تعداد طبقه‌ها را تعیین می‌کند، در صورتی که در روش‌های ارائه‌شده‌ی قبلی اقلام باید در سه طبقه گروه‌بندی می‌شدند. دوم اینکه هدف اصلی روش ارائه‌شده در این مقاله تعیین نوع طبقه نیست، بلکه فقط تعداد طبقه‌ها را به‌دست می‌آورد. سوم اینکه یکی از مهم‌ترین اهداف مدیران موجودی (حداقل‌کردن مجموع هزینه‌های سفارش‌دهی و نگهداری) در این مقاله برآورده می‌شود. همچنین نرخ گردش موجودی‌ها را نیز مد نظر قرار می‌دهد. با توجه به اینکه در خصوص نوع و تعداد طبقه‌های اقلام شاخص‌های کیفی و کمّی زیادی می‌توانند تأثیرگذار باشند، نتایج این روش می‌تواند کمک فراوانی به مدیران باتجربه‌ی موجودی بکند، زیرا به جای ارائه‌ی یک نتیجه‌ی منحصربه‌فرد، مجموعه‌ای از جواب‌های مختلف را به تصمیم‌گیرنده ارائه می‌دهد.

در این مقاله سعی شده است یکی از جدیدترین این راه‌ها آزموده شود. در این روش با استفاده از الگوریتم بهینه‌سازی جمعی راه‌حل‌های چندهدفه(MOPSO) که نوعی الگوریتم تکاملی و همگراست، به‌دنبال تعیین تعداد طبقات موجودی هستیم. در این راستا، با توجه به اهداف حداقل‌کردن هزینه‌های انبار و سفارش‌دهی و افزایش نرخ گردش موجودی، تعداد طبقات تعیین می‌شود. پس از نوشتن برنامه‌ی رایانه‌ای الگوریتم آن را آزمودیم و نتایج مطلوب و قابل ملاحظه‌ای به‌دست آمد. نتایج به‌دست‌آمده به‌دلیل در نظر گرفتن توأمان دو عامل مهم در تصمیم‌گیری مدیریت، یعنی هزینه‌های نگهداری و سفارش‌دهی و نرخ گردش موجودی، از قابلیت اتکای

بسیار زیادی برخوردارند.

از آنجا که این روش در تعیین تعداد طبقات موجودی بسیار نو بوده و هنوز در ایران به آن پرداخته نشده است، پیشنهاداتی بدین‌شرح ارائه می‌شود:

به‌دلیل گستردگی موجودی‌ها، پیشنهاد می‌شود از روش ارائه‌شده در تحقیق برای طبقه‌بندی مواد اولیه، کالاهای در جریان ساخت و محصولات نهایی به‌طور هم‌زمان استفاده شود؛ یعنی همبستگی تقاضای بین تمامی اقلام موجودی نیز وارد الگوریتم شود.

در این مقاله کارایی این الگوریتم از جنبه‌ی تئوریک اثبات شد. پیشنهاد می‌شود کارهای آینده بر روی بررسی میدانی آن متمرکز شود.

در الگوریتم این مقاله برخی از پارامترها را می‌توان با استفاده از روش‌های دیگر به‌دست آورد که در بخش ادبیات بیان شده است. پیشنهاد می‌شود روش‌های گفته‌شده در این الگوریتم ادغام شده و نتایج آنها با نتایج به‌دست‌آمده از این تحقیق مقایسه شود.

از MOPSO می‌توان در مسائل مختلفی چون طبقه‌بندی پایگاه‌های داده نیز استفاده کرد. پیشنهاد می‌شود پژوهشگران علاقه‌مند در این زمینه نیز وارد شوند و به‌دنبال حل مسائل از روش‌هایی باشند که دیگران به آن توجهی نداشته‌اند.

منابع و مآخذ

[۱] صفایی، عبدالحمید؛ مدهوشی، مهرداد؛ اسماعیل‌زاده، منصور، **تلفیق دو مدل طبقه‌بندی ABC چندمعیاره‌ی موجودی**؛ فصلنامه‌ی مطالعات مدیریت، شماره‌ی ۵۷، ۱۳۸۷، صص۱۳۳-۱۴۹.

[۲] رضایی، جعفر؛ اسماعیل‌زاده، منصور، **مقایسه‌ی روش‌های مختلف طبقه‌بندی ABC چندمعیاره‌ی موجودی**؛ فصلنامه‌ی علمی ـ پژوهشی مطالعات مدیریت صنعتی، شماره‌ی ۱۷، ۱۳۸۸، صص ۱-۲۲.

[3] Weihrich, H., Koontz, H., Management: A Global Perspective, McGraw-hill, 11th Edition, 2006.

[4] Guvenir, H.A., Erel, E., Multi criteria inventory classification using a genetic algorithm, European Journal of Operational Research, vol. 105, 1998. pp. 29-37.

[5] Silver, E.A., Pyke, D.F., Peterson, R., Inventory Management and Production Planning and Scheduling, John Wiley & Sons, New York, Third ed., 1998.

[6] Cohen, M.A., Ernst, R., Multi-item classification and generic inventory stock control policies, Production and Inventory Management Journal, vol. 29(3), 1988. pp. 6-8.

[7] Partovi, F.Y., Burton, J., Using the analytic hierarchy process for ABC analysis, International Journal of Productions Management, vol. 13(9), 1993. pp. 29-44.

[8] Flores, B.E., Whybark, D.C., Implementing multiple criteria ABC analysis, Journal of Opration Management, vol. 7(1), 1987. pp. 79-84.

[9] Partovi, F.Y., Anandarajan, M., Classification inventory using an artificial neural network approach, Computer & industrial Engineering, vol. 41, 2005. pp. 389-404.

[10] Kennedy, J., Eberhart R.C., Particle swarm optimization. In: Proceedings of the IEEE conference on neural networks, Perth, Australia, 1995. pp. 1942–8.

[11] Eberhart, R.C., Kennedy, J., A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, IEEE Press, Piscataway, NJ, 1995. pp. 39–43.

[12] Eberhart, R.C., Shi, Y., Computational Intelligence: Concepts to Implementations, Morgan Kaufmann, 2003.

[13] Zhu, H., et al., Particle Swarm Optimization (PSO) for the constrained portfolio optimization problem, Expert Systems with Applications, 2011, doi:10.1016/j.eswa.2011.02.075.

[14] Omkar, S.N., et al., Quantum behaved Particle Swarm Optimization (QPSO) for multi objective design optimization of composite structures, Expert Systems with Applications, vol. 36, 2009. pp. 11312–22.

[15] Tsai, C.Y., Yah, S.W., A multiple objective particle swarm optimization approach for inventory classification, Int. J. Production Economics, vol. 114, 2008. pp. 656–66.

[16] Feng, Y., Zheng, B., Li. Z., Exploratory study of sorting particle swarm optimizer for multi objective design optimization, Mathematical and Computer Modelling, vol. 52, 2010. pp. 1966-75.

[17] Li, X., et al., A nondominated sorting particle swarm optimizer for multi objective optimization, in: Proceedings of the Genetic and Evolutionary Computation, Springer, 2003. pp. 37–48.

[18] Raquel, C., Naval, P. Jr., An effective use of crowding distance in multi objective particle swarm optimization, in: Proceedings of the Conference on Genetic and Evolutionary Computation, ACM Press, New York, NY, USA, 2005. pp. 257–264.
[19] Fieldsend, J., Singh, S., A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence, in: Proceedings of The UK Workshop on Computational Intelligence, UK, 2002. pp. 34–44.
[20] Xiaohui, H., Eberhart, R., Multi objective optimization using dynamic neighborhood particle swarm optimization, in: Proceedings of the Congress on Evolutionary Computation, vol. 2, 2002. pp. 1677–81.
[21] Pulido, G., Coello, C., Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer, in: Proceedings of the Genetic and Evolutionary Computation Conference, Springer, 2004. pp. 225–37.
[22] Liu, D., et al., A multi objective memetic algorithm based on particle swarm optimization, IEEE Transactions on Systems, Man and Cybernetics, Part B 37 (1), 2007. pp. 42–50.
[23] Liu, D., et al., On solving multi objective bin packing problems using evolutionary particle swarm optimization, European Journal of Operational Research, 2008. pp. 357–82.
[24] Goldberg, D.E., Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[25] Fogel, L.J., Marsh, A.J., Walsh, M.J., Artificial Intelligence through Simulated Evolution, Wiley & Sons, New York, 1966.
[26] Reynolds, C.W., Flocks, herds and schools: a distributed behavioral model, Comput, Graphics, vol. 21 (4), 1987. pp. 25–34.
[27] Heppner, F., Grenander, U., A stochastic nonlinear model for coordinated bird flocks, in: S. Krasner (Ed.), The Ubiquity of Chaos, AAAS Publications, Washington, DC, 1990.
[28] Wilson, E.O., Sociobiology: The New Synthesis, Belknap Press, Cambridge, MA, 1975.
[29] Suganthan, P.N., Particle swarm optimizer with neighborhood operator. In: Proceedings of the 1999 congress of evolutionary computation, vol. 3. IEEE Press, 1999. pp. 1958–62.
[30] Shi, Y., Eberhart, R., Parameter selection in particle swarm optimization, in: Proceedings of Evolutionary Programming, 1998. pp. 591–600.
[31] Alatas, B., Akin, E., Ozer, A.B., Chaos embedded particle swarm optimization algorithms, Chaos, Solitons and Fractals, vol. 40, 2009. pp. 1715–34.
[32] Shi, Y., Eberhart, R.C., A modified particle swarm optimizer, IEEE IntConfComputIntell, 1998. pp. 69–73.
[33] Shi, Y., Eberhart, R.C., Empirical study of particle swarm optimization. In: Proceedings of the 1999 IEEE congress on evolutionary computation, Piscataway (NJ): IEEE Press, 1999. pp. 1945–50.
[34] Zheng, Y.L., etal., On the convergence analysis and parameter selection in particle swarm optimization, In: Proceedings of the 2003 IEEE international conference on

machine learning and cybernetics, Piscataway (NJ): IEEE Press, 2003. pp. 1802–7.
[35] Zheng, Y.L., et al., Empirical study of particle swarm optimizer with an increasing inertia weight, In: Proceedings of the 2003 IEEE congress on evolutionary computation, Piscataway (NJ): IEEE Press, 2003. pp. 221–6.
[36] Jiao, B., et al., A dynamic inertia weight particle swarm optimization algorithm, Chaos, Solitons& Fractals, vol. 37(3), 2008. pp. 698–705.
[37] Zhang, L., Yu, H., Hu, S., A new approach to improve particle swarm optimization, GECCO 2003, LNCS, vol. 2723, 2003. pp. 134–9.
[38] Eberhart, E., Shi, Y., Tracking and optimizing dynamic systems with particle swarms. In: Proceedings of the 2001 IEEE congress on evolutionary computation, Piscataway (NJ): IEEE Press, 2001. pp. 94–100.
[39] Park, J.B., et al., An improved particle swarm optimization for economic dispatch with valve-point effect, Int J Innov Energy System Power, vol. 1(1), 2006. pp. 1–7.
[40] Jiang, C., Etorre, B.A., Hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimization, Math Compute Simul, vol. 68, 2005. pp. 57–65.
[41] Ratnaweera, A., Halgamure, S.K., Watson, H.C., Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, IEEE Trans Evol Compute, vol. 8 , 2004. pp. 240–55.
[42] Carlisle, A., Dozier, G., An off the-Shelf PSO, In: The particle swarm optimization workshop, 2001. pp. 1–6.
[43] Coelho, L.S., Mariani, V.C., A novel chaotic particle swarm optimization approach using He'non map and implicit filtering local search for economic load dispatch, Chaos, Solitons& Fractals, vol. 39(2), 2009. pp. 510–8.
[44] Chuanwen, J., Bompard, E.A., Self-adaptive chaotic particle swarm algorithm for short term hydroelectric system scheduling in deregulated environment, Energy Converse Manage, vol. 46, 2005. pp. 2689–96.
[45] Chakravarty, A.K., Multi-item inventory grouping whit dependent set-up cost and group overhead cost, Engineering Cost and Production Economics, 1986. pp. 13-23.
[46] صفایی، عبدالحمید؛ اسماعیل‌زاده، منصور، **ارائه‌ی رهیافتی جدید برای مقایسه‌ی نتایج به‌کارگیری مدل‌های طبقه‌بندی ABC چندمعیاره‌ی موجودی (مطالعه‌ی موردی: شرکت سایپا)**، دوماهنامه‌ی علمی ـ پژوهشی دانشگاه شاهد، سال هجدهم شماره‌ی ۲-۴۷، تیرماه ۱۳۹۰، صص۲۰۷-۲۲۴.