مهندسی صنایع و مدیریت شریف
دوره‌ی ۱ - ۲۷، شماره‌ی ۱، ص. ۱۱۳-۱۲۵

انتخاب سبد سهام بااستفاده از بهینه‌سازی گروه ذره‌ها

حمیدرضا گل‌مکانی (استادیار)
مهرشاد فاضل (کارشناس ارشد)
دانشکده‌ی مهندسی صنایع، دانشگاه تفرش

در این نوشتار الگوریتمی ترکیبی بر مبنای روش «بهینه‌سازی گروه ذره‌ها» برای حل مدل توسعه یافته‌ی میانگین-واریانسِ انتخاب سبد سهام ارائه می‌شود. اغلب، برای تطابق مدل اولیه میانگین-واریانس با دنیای واقعی، محدودیت‌های مختلفی به آن افزوده می‌شود. در این نوشتار، چهار نوع محدودیت، محدودیتِ کاردینالیتی (شامل محدودیت روی نسبت هر سهم در سبد و تعداد (نوع) سهم در سبد)، حداقل مقدار معامله و ارزش بخش لحاظ می‌شوند. با استفاده از سه گروه از داده‌های نمونه، نتایج اجرای الگوریتم جدید موسوم به CBIPSO [1] (ترکیب بهینه‌سازی گروه ذره‌های ارتقاء یافته و دودویی) و یک الگوریتم ژنتیک، براساس چهار معیار، مورد مقایسه قرار می‌گیرند. معیارهای مورد ارزیابی عبارت‌اند از سرعت رسیدن به جواب، میانگین جواب‌های به‌دست آمده، بهترین جواب حاصل در چند اجرای متوالی الگوریتم و انحراف معیار جواب‌ها. همچنین تحلیل حساسیت عملکرد CBIPSO با تغییر پارامترهای مختلف در الگوریتم و مسئله‌بررسی می‌شود. نتایج تحقیق نشان می‌دهند که CBIPSO به‌طور کلی، و به‌ویژه وقتی که تعداد سهم‌های در دسترس و یا تعداد سهم در سبد زیاد می‌شود، بهتر از الگوریتم ژنتیک، عمل می‌کند.

واژگان کلیدی: انتخاب سبد سهام، حداقل مقدار معامله، محدودیت‌های کاردینالیتی، ارزش‌بخش، بهینه‌سازی گروه ذره‌ها.

golmakni@mie.utoronto.ca
mshdfazel@gmail.com

۱. مقدمه

سرمایه‌گذاران همواره به دنبال یافتن سبد یا ترکیبی از سهم‌های مختلف‌اند که ضمن افزایش بازده، ریسک سرمایه‌گذاری را در پایین‌ترین حد نگه دارد. در سال ۱۹۵۲ مدل میانگین-واریانس برای انتخاب سبد سهام ارائه شد.[۱] در این مدل، واریانس یا انحراف معیار بازده هر سهم نشان‌گر مقدار ریسک سرمایه‌گذاری روی آن سهم است. این ریسک به میزان ارتباط سهم‌ها با یکدیگر ــ که از طریق کوواریانس بین بازده سهم‌ها تعیین می‌شود ــ بستگی دارد. طراح این مدل (مارکوویتس) روش خط بحرانی را برای حل مدل خود به کار بست[۱و۲] اما، این روش در مواردی که تعداد سهم‌های در دسترس زیاد می‌شوند کارایی ندارد.[۳] همچنین با افزوده‌شدن محدودیت‌های مختلف به مدل، پیچیدگی مدل افزایش می‌یابد و حل مسئله با روش‌های معمول مقدور نخواهد بود. مثلاً ثابت شده است که مسئله‌ی مارکوویتس همراه با محدودیت‌های کاردینالیتی و کم‌ترین مقدار معامله، مستقل از نوع تابع هدف (تابع ریسک)، یک مسئله‌ی NP-hard است.[۴] به‌این معنا که الگوریتم کارآمدی برای حل قطعی این مسئله در ابعاد بزرگ وجود ندارد. به‌این ترتیب، روش‌های ابتکاری و فرا‌ابتکاری مختلفی برای حل مدل‌های تکمیل‌یافته‌ی مارکوویتس استفاده شده‌اند.

محققین با واردکردن هزینه‌ی معامله در مدل مارکوویتس، اقدام به حل آن با استفاده از روش برنامه‌ریزی تصادفی کردند.[۵] همچنین مسئله‌ی انتخاب سبد (بدون محدودیت) را با استفاده از الگوریتم ژنتیک حل کردند.[۶] در پژوهشی دیگر، الگوریتم ژنتیک برای حل مسئله‌ی مارکوویتس بدون درنظرگرفتن محدودیت‌های عملی رایج به کار بستند[۷] و صرفه‌جویی زمانی حاصل را اثبات کردند. در سال ۱۹۹۶،[۸] با اتخاذ مدلی مشابه مدل یاشیموتو[۹و۱۰] به حل این مدل با استفاده از جست‌وجوی ممنوع پرداختند. اقدام محققین در سال ۲۰۰۰ نقطه‌ی عطفی در حل مسئله‌ی مارکوویتس همراه با محدودیت‌های کاردینالیتی است.[۱۱] آنها با ایجاد ناپیوستگی در مرز کارا درنتیجه‌ی اضافه‌شدن محدودیت‌های گسسته (مانند محدودیت‌های کاردینالیتی) به مدل اشاره و مدل درجه دوِ حاصل را از نظر محاسباتی بررسی کردند. سپس، سه روش فرا‌ابتکاری مبتنی بر الگوریتم ژنتیک، جست‌وجوی ممنوع و سردشدن شبیه‌سازی شده‌ی فلزات برای حل مدل ارائه کردند. مدل مورد نظر آن‌ها همان مسئله‌ی برنامه‌ریزی درجه دوِ صحیح ترکیبی (MIQP)[۲] است که از ورود محدودیت‌های کاردینالیتی در مدل اولیه‌ی مارکوویتس حاصل می‌شود.

در سال ۲۰۰۰ یک الگوریتم ژنتیک و یک روش مبتنی بر جست‌وجوی ممنوع برای حل مدلی با محدودیت‌های مختلف (شامل محدودیت‌های کاردینالیتی و هزینه‌های معامله‌ی غیرخطی) پیشنهاد شد[۱۲] و در کنار آن، ورود محدودیت‌های متنوع دیگر در مدل ــ ازجمله محدودکردن سرمایه‌گذاری در بخش‌های مختلف ــ برای تحقیقات آتی پیشنهاد شد.

تاریخ: دریافت ۱۳۸۸/۷/۲۵، اصلاحیه ۱۳۸۹/۶/۳۱، پذیرش ۱۳۸۹/۷/۲۰.

در سال ۲۰۰۱ دو شیوه‌ی ابتکاری برای حل سه مدل به کار گرفته شد؛[۱۳] این مدل‌ها همان مدل مارکوویتس است که جداگانه با محدودیت‌های سطح خرید، کاردینالیتی و کم‌ترین مقدار معامله لحاظ شده است. نتایج حاصل از به‌کارگیری این دو شیوه با نتایج حاصل از تحقیقاتی که به طرح مدلMIQP انجامید[۱۱] مقایسه شد. در سال ۲۰۰۲ روش جست‌وجوی ممنوع برای حل مسئله‌ی انتخاب سبد سهام به‌کار گرفته شد.[۱۴] پیش از آن نیز با ترکیب مفاهیم سردشدن شبیه‌سازی شده‌ی فلزات و راهکارهای تکاملی و ابداع یک روش جست‌وجوی محلی، نسبت به حل مدل مارکوویتس با محدودیت‌های کاردینالیتی اقدام شد.[۱۵]

محققین کارایی الگوریتم‌های ژنتیک ابداع‌شده در گذشته (نظیر NSGA و NSGA-II [۳] در حل مدل مارکوویتس) را مورد بررسی قرار داده‌اند.[۱۶] مدل مورد نظر آن‌ها محدودیت‌های کاردینالیتی را نیز شامل می‌شد.

در سال ۲۰۰۳ ابتدا مدلی در نظر گرفته شد که دارای محدودیت‌های کاردینالیتی، گردش[۴] شامل خرید و فروش (معادل با محدودیت هزینه‌ی معامله) و دادوستد[۵] (معادل با محدودیت کمینه مقدار معامله) است؛[۱۷] سپس این مدل به‌کمک روش سردشدن شبیه‌سازی شده‌ی فلزات حل شد.

در سال ۲۰۰۴، بااستفاده از الگوریتم تکاملی چندمنظوره[۶] مسئله‌یی که فقط واجد محدودیت تعداد (نوع) سهم است، حل شد.[۱۸] با بهره‌گیری از این الگوریتم‌های تکاملی و چندمنظوره، نسبت به حل مسئله‌ی مارکوویتس با محدودیت‌های کاردینالیتی و کم‌ترین مقدار معامله اقدام شد.[۱۹-۲۱]

در سال ۲۰۰۵ مدل مارکوویتس با محدودیت‌های مختلف مورد بررسی قرار گرفت.[۲۲] در این مطالعات، علاوه بر محدودیت‌های کاردینالیتی، محدودیت دیگری به‌نام محدودیت ۵-۱۰-۴۰[۷] نیز بررسی، و در نهایت نتایج حاصل از مقایسه‌ی روش‌های حل ابتکاری و فراابتکاری پیشنهاد شده برای مدل‌های مورد نظر ارائه شد.

در سال ۲۰۰۶ محققین برای انتخاب سبد سهام ارزش بخش را مبنا قرار دادند و الگوریتم ژنتیک خود را برای حل مسئله ارائه کردند.[۲۳] البته، مسئله‌ی آن‌ها مدل مارکوویتس نبود، بلکه از روش تعقیب شاخص[۸] (مدل تک‌عاملی شارپ) استفاده کردند. در تحقیقی جدا، ضمن معرفی محدودیت ارزش بخش، یک الگوریتم ژنتیک ابتکاری برای حل مدل حاصل پیشنهاد کردند.[۲۴و۲۵] و نشان دادند که کارایی الگوریتم ژنتیک ابتکاریِ پیشنهادی از روش‌های دیگر بیشتر است. در ادامه‌ی مطالعاتِ آنان یک الگوریتم ژنتیک وفقی برای حل مدل مارکوویتس با محدودیت نسبت هر سهم در سبد ارائه شد.[۲۶]

در سال ۲۰۰۷، شش مدل مبتنی بر مدل مارکوویتس که واجد محدودیت کم‌ترین مقدار معامله بودند بررسی شد و الگوریتم ژنتیک خاصی برای حل آن‌ها ارائه گردید.[۲۷] در همین سال یک شبکه‌ی عصبی مصنوعی Hopfield برای حل مسئله‌ی مارکوویتس همراه با محدودیت‌های کاردینالیتی تشریح شد.[۲۸] در سال ۲۰۰۷ برای نخستین‌بار گونه‌یی از الگوریتم بهینه‌سازی گروه ذره‌ها به‌منظور حل مسئله‌ی مارکوویتس مورد استفاده قرار گرفت.[۲۹] این الگوریتم مسئله‌ی MIQP را شامل نمی‌شد زیرا، در آن محدودیت‌های کاردینالیتی (محدودیت تعداد یا نوع سهم) وجود نداشت. همچنین در این مدل کم‌ترین مقدار معامله در نظر گرفته نشد و از این‌رو فضای جست‌وجو در آن فضای اعداد حقیقی بود. در این مدل اگر چه هزینه‌ی معامله ازجمله مالیات وارد شد و محدودیت نسبت هر سهم نیز در نظر گرفته شد، مسئله‌ی چندان پیچیده‌یی مطرح نبود.

در سال ۲۰۰۸ یک سیستم ایمنی مصنوعی برای حل مدل مارکوویتس همراه با محدودیت‌های کاردینالیتی (و بدون درنظر گرفتن محدودیت‌های ارزش‌بخش) پیشنهاد،[۳۰] و با سایر الگوریتم‌ها مقایسه شد.

علاوه بر تحقیقات یادشده، مطالعات وسیعی در حوزه‌ی مسئله‌ی تخصیص دارایی[۹]، به‌ویژه مسئله‌ی مارکوویتس در تعریف ریسک و متدهای مختلف اندازه‌گیری آن صورت گرفته است که از آن جمله می‌توان به استفاده از روش‌های قابلیت اطمینان[۱۰] در تحلیل عدم قطعیت[۳۱] اشاره کرد.

الگوریتم بهینه‌سازی گروه ذره‌ها در بسیاری از مسائل بهینه‌سازی مورد استفاده قرار گرفته است. با استفاده از آزمون‌های متعدد، بین الگوریتم ژنتیک، الگوریتم بهینه‌سازی گروه ذره‌ها و سه الگوریتم دیگر برحسب معیارهای نرخ موفقیت، کیفیت جواب و زمان پردازش، مقایسه‌یی صورت گرفته است[۳۲] که نشان می‌دهد بهینه‌سازی گروه ذره‌ها از نظر نرخ موفقیت و کیفیت جواب در رده‌ی نخست، و از نظر زمان در رده‌ی دوم قرار دارد. مقایسه‌ی دیگری نیز بین بهینه‌سازی گروه ذره‌ها و الگوریتم ژنتیک انجام شده[۳۳] که در مجموع، برتری الگوریتم بهینه‌سازی گروه ذره‌ها را نسبت به الگوریتم ژنتیک نشان می‌دهد.

در این نوشتار چهار نوع محدودیت ــ محدودیتِ کاردینالیتی (شامل محدودیت روی نسبت هر سهم در سبد و تعداد یا نوع سهم در سبد)، کم‌ترین مقدار معامله و ارزش بخش برای مسئله ــ لحاظ، و الگوریتمی فراابتکاری مبتنی بر الگوریتم بهینه‌سازی گروه ذره‌ها برای حل آن ارائه می‌شود و نتایج حاصل با نتایج اجرای الگوریتم ژنتیک مقایسه خواهد شد. معیارهای مورد ارزیابی عبارت‌اند از: سرعت رسیدن به جواب، میانگین جواب‌های به‌دست آمده، بهترین جواب حاصل در چند اجرای متوالی الگوریتم، و انحراف معیار جواب‌ها. نتایج تحقیق نشان می‌دهند که کارایی الگوریتم پیشنهادی، به‌طورکلی ــ به‌ویژه وقتی تعداد سهم‌های در دسترس یا تعداد سهم در سبد زیاد می‌شود ــ درخصوص مجموعه‌ی اطلاعاتی مورد استفاده بیشتر از الگوریتم ژنتیک است.

در ادامه، ابتدا در بخش ۲ مدل اولیه و مدل توسعه‌یافته ارائه خواهد شد و سپس در بخش ۳ روش بهینه‌سازی گروه ذره‌ها توضیح داده خواهد شد. در بخش ۴ به معرفی الگوریتم ابتکاری مبتنی بر بهینه‌سازی گروه ذره‌ها پرداخته شده است. بخش ۵ نیز به بررسی نتایج تجربی اختصاص دارد و در بخش ۶ نتیجه‌گیری کلی ارائه شده است.

۲. مدل انتخاب سبد سهام

مدل اولیه‌ی مارکوویتس عبارت است از:[۱۱]

$$\min \ \sigma^2_{R_p} = \sigma^2_p = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j \, Cov(\bar{R}_i, \bar{R}_j) \qquad (1)$$

مشروط بر این که:

$$\bar{R}_p = E(R_p) = \sum_{i=1}^{N} w_i \bar{R}_i \geq R \qquad (2)$$

$$\sum_{i=1}^{N} w_i = 1 \qquad (3)$$

$$w_i \geq 0, \qquad \forall i \in \{1, 2, ..., N\} \qquad (4)$$

که در آن $\bar{R}_i$ بازده مورد انتظار سهم iام، w_i نسبت سهم iام در سبد، N تعداد سهم‌های در دسترس، R کم‌ترین بازده قابل قبول سبد برای سرمایه‌گذار و $Cov(\bar{R}_i, \bar{R}_j)$ کوواریانس بین بازده مورد انتظار سهم‌های i و j است. این مدل بیان می‌دارد که به‌ازاء کم‌ترین مقداری از بازده، ریسک سرمایه‌گذاری باید کمینه شود.

برآورده شدن کمینه بازده مورد انتظار سرمایه‌گذار به‌صورت محدودیت ۲ لحاظ شده است. رابطه‌ی ۳ بیان‌گر محدودیت بودجه است که طبق آن مجموع نسبت‌های سرمایه‌گذاری باید برابر با ۱ باشد و مفهوم آن اختصاص کل بودجه به سرمایه‌گذاری است. محدودیت ۴ تضمین می‌کند که نسبت‌های سرمایه‌گذاری عددهای نامنفی باشند.

با افزودن محدودیت‌های کم‌ترین مقدار معامله، کاردینالیتی و ارزش بخش به مدل تکامل‌یافته‌ی مورد نظر خود می‌رسیم که بیانِ آن چنین است:

$$\min \sigma^2_{R_p} = \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j Cov(\bar{R}_i, \bar{R}_j) \qquad (۵)$$

که در آن:

$$w_i = \frac{x_i c_i z_i}{\sum_{j=1}^{N} x_j c_j z_j} \qquad (۶)$$

و

$$\sum_{i=1}^{N} z_i = M \le N, \qquad M, N \in \mathbb{N}, \ \forall i = 1, ..., N \ z_i \in \{0, 1\} \qquad (۷)$$

به‌گونه‌یی که:

$$\sum_{i=1}^{N} x_i c_i z_i \bar{R}_i \ge BR \qquad (۸)$$

$$\sum_{i=1}^{N} x_i c_i z_i \le B \qquad (۹)$$

$$0 \le B_{Low_i} \le x_i c_i \le B_{Up_i} \le B \qquad (۱۰)$$

$$\sum_{i_s} W_{i_s} \ge \sum_{i_{s'}} w_{i_{s'}} \qquad \forall y_s, y_{s'} \ne 0; \ \ s, s' \in \{1, ..., s\}, \ \ s < s' \qquad (۱۱)$$

که در آن:

$$y_s = \begin{cases} 1 & if \sum_{i_s} z_i > 0 \\ 0 & if \sum_{i_s} z_i = 0 \end{cases} \qquad (۱۲)$$

و $i_s, i_{s'} \in \{1, ..., N\}$.

■

متغیرها و ثابت‌های به کار رفته در رابطه‌های بالا عبارت‌اند از:
B: بودجه‌ی قابل تخصیص؛ c_i: حاصل ضرب قیمت سهم iام در کم‌ترین مقدار معامله؛ x_i: تعداد c_iها (عدد صحیح)؛ z_i: متغیر صفر و یک که بودن یا نبودن سهم iام در سبد را تعیین می‌کند؛ M: تعداد سهم‌های مجاز در سبد = اندازه‌ی سبد؛ B_{Low_i}: کمینه بودجه‌ی قابل سرمایه‌گذاری روی سهم iام؛ B_{Up_i}: بیشینه بودجه‌ی قابل سرمایه‌گذاری روی سهم iام؛ i_s: اندیس سهمی که متعلق به بخش sام است؛ δ: تعداد کل بخش‌ها برای N شرکت (سهم)؛ و y_s: متغیر صفر و یک که انتخاب‌شدن یا نشدن سهمی از بخش sام را در سبد نشان می‌دهد.

در بازار واقعی، خرید و فروش سهم‌ها به هر تعداد دلخواه ممکن نیست. مثلاً هرگز نمی‌توان ۲ سهم از سهام یک شرکت را معامله کرد بلکه، معامله‌ی سهم‌ها به‌صورت بسته‌هایی با تعداد مشخص صورت می‌پذیرد. به‌عنوان مثال ضریبی از ۱۰۰ سهم یک شرکت را می‌توان خرید و فروش کرد. این محدودیت «کم‌ترین مقدار معامله» نام دارد. برای واردکردن این محدودیت در مدل، متغیر c_i را تعریف می‌کنیم. متغیر x_i تعداد c_iها را تعیین می‌کند. در واقع، هدف ما در حل مسئله‌یافتن x_iها و درنتیجه z_iهای مناسب است. حاصل ضرب $x_i c_i$ مقدار بودجه‌ی سرمایه‌گذاری شده روی سهم iام را مشخص می‌کند.

محدودیت‌های کاردینالیتی، اِعمال محدودیت بر اندازه‌ی سبد (تعداد سهم‌های درون سبد) و نسبت هر سهم در سبد را شامل می‌شود. محدودیت نخست با تعریف متغیر صفر و یک z_i برای هر سهم i، همراه با محدودیت ۷ بیان می‌شود. یک کارکرد مهم این محدودیت محدودکردن هزینه‌ی معامله است زیرا، بر اثر محدودشدن تعداد سهم‌هایی که باید خریداری شوند، تعداد معاملاتِ لازم نیز محدود می‌شود. همچنین، از اختصاص بودجه‌ی ناچیز به خرید تعداد زیادی سهم مختلف جلوگیری می‌شود. از طرفی، برای جلوگیری از اختصاص همه‌ی بودجه به یک یا چند سهم کم، این محدودیت به جای کوچک‌تر یا مساوی به‌صورت برابری تعریف می‌شود. متغیر z_i در رابطه‌های ۶، ۸، ۹ و ۱۲ وارد می‌شود تا فقط تعداد معینی (M) از کل (N) سهمِ در دسترس در محاسبات دخیل شوند.

درخصوص حد بالا (B_{Up_i}) و حد پایین (B_{Low_i}) بودجه‌ی قابل تخصیص به سهم i، اگر محدودیت ۱۰ مثلاً به‌دلیل $B_{Up_i} \ge B$ برقرار نباشد، آنگاه این احتمال هست که در فرایند حل مسئله، همه‌ی بودجه فقط به یک سهم اختصاص یابد که مطلوب نیست. حدهای بالا و پایین سرمایه‌گذاری روی یک سهم از مطالعات تاریخی روی حجم معامله‌های صورت گرفته‌ی آن سهم قابل دست‌یابی است. درصورتی که چنین اطلاعاتی در دسترس نباشد، می‌توان حد بالا (B_{Up_i}) را با رابطه‌ی (قیمت سهم iام/B) برابر قرار داد.[۳۴]

طبیعی است که بیشینه مقدار عبارت $\sum_{i=1}^{N} x_i c_i z_i$، با بودجه‌ی سرمایه‌گذار برابری کند. بنابراین، محدودیت بودجه طبق رابطه‌ی ۹ برقرار می‌شود. اگر این رابطه را به‌صورت $1 \le \frac{\sum_{i=1}^{N} x_i c_i z_i}{B}$ بنویسیم، به ارتباط نزدیک آن با رابطه‌ی ۳ در مدل اولیه‌ی مارکوویتس پی می‌بریم.

در این‌جا محدودیتِ ۲ به‌صورت رابطه‌ی ۸ بازنویسی شده است. سمت راست این رابطه نشان‌گر کم‌ترین سود ریالی مورد انتظار از کل بودجه، و سمت چپ آن نشان‌دهنده‌ی سود حاصل از سرمایه‌گذاری در سبد است.

محدودیت دیگر، محدودیت ارزش[۱۲] است. براساس این محدودیت، در M سهم انتخاب‌شده از میان N سهمِ در دسترس، به شرکت‌های وابسته به بخش‌های دارای گردش مالی بیشتر، سرمایه‌ی بیشتری اختصاص داده می‌شود. در بازار سرمایه رده‌بندی[۱۳]های مختلفی روی شرکت‌های حاضر در بازار ــ به‌ویژه براساس نوع فعالیت آنها ــ صورت می‌پذیرد. این رده‌بندی‌ها با عنوان‌هایی نظیر صنعت، بخش[۱۴] و... بیان می‌شوند که از آن جمله می‌توان به صنعت نفت‌وگاز، خودرو، فولاد و... اشاره کرد. اگر ارزش و حجم معامله‌های انجام‌شده در یک رده بیشتر از رده‌های دیگر باشد، سهم‌های درون آن رده عملکرد بهتری خواهند داشت[۲۴] و درنتیجه بهتر است سهم بیشتری از سرمایه به سهم‌های رده‌های با ارزش‌تر اختصاص یابد. البته ممکن است عملکرد یک شرکت در رده‌ی باارزش‌تر بازار، بدتر از عملکرد شرکت‌های واقع در رده‌ی کم‌ارزش‌تر باشد، مثلاً اگر فرض کنیم که ارزش معامله‌های سهام بخش شیمیایی و پتروشیمی بیش از بخش خودرو باشد، ممکن است عملکرد شرکت پتروشیمی «ش» از عملکرد همه‌ی شرکت‌های بخش خودرو بدتر باشد. از این رو، طبیعی است که سرمایه‌گذاری روی چنین شرکتی در اولویت قرار نگیرد.

بنابراین، محدودیت ارزش بخش به جای آن که روی سهام N شرکت در دسترس

اعمال شود، پس از انتخاب سهم‌ها (شرکت‌ها) روی سهام M شرکت انتخاب‌شده اعمال می‌شود. به‌این ترتیب شرکت پتروشیمی «ش» از اولویت انتخاب کنار گذاشته می‌شود. برای این کار، متغیر صفر و یک y_s تعریف می‌شود. طبق رابطه‌ی ۱۲ این متغیر برای بخش sام زمانی مقدار ۱ به خود می‌گیرد که دست‌کم یک سهم از آن بخش در سبد قرار گرفته باشد. در غیر این صورت مقدار آن صفر می‌شود.

این محدودیت در برخی از منابع عنوان «محدودیت رده»[15] گرفته، اما در مدل وارد نشده است.[11] این محدودیت برای نخستین‌بار در مدل انتخاب سبد سهام وارد شده و به کار گرفته شده است.[24] در این‌جا برای بیان محدودیت ارزش از رابطه‌ی ۱۱ استفاده می‌شود که از دیگر رابطه‌های به‌کار گرفته شده در دیگر مطالعات[24] دقیق‌تر است. همچون دیگر مطالعات، فرض می‌شود که بخش ۱ با ارزش‌ترین و بخش δام کم‌ارزش‌ترین بخش است و بخش‌ها از نظر ارزش به‌ترتیب نزولی مرتب می‌شوند. فرض کنید که از دو بخش s و s' $(s > s')$، سهم‌هایی در سبد قرار گرفته باشد (یعنی $y_s, y_{s'} \neq 0$). در این‌صورت رابطه‌ی ۱۱ بیان می‌دارد که مجموع ارزش‌های بازار سهم‌های انتخابی از بخش sام، از مجموع ارزش‌های بازار سهم‌های انتخابی از بخش s'ام بیشتر است.

۳. بهینه‌سازی گروه ذره‌ها

بهینه‌سازی گروه ذره‌ها (PSO)[16]، یک الگوریتم بهینه‌سازی است[35-37] که در آن ذره‌ها و به‌عبارت دقیق‌تر موقعیت[17] آن‌ها جواب‌های بالقوه‌یی هستند که در فضای جست‌وجو به پرواز درمی‌آیند و تا رسیدن به یک معیار مشخص، با محاسبه‌ی یک رابطه‌ی تناسب به دنبال بهترین مقدار تابع هدف می‌گردند. بنابراین، ابتدا جمعیتی از ذره‌ها یا موقعیت‌های اولیه تشکیل می‌شود. سپس طبق رابطه‌های خاص و در گام‌های زمانیِ پیاپی، در فضای جست‌وجو به گردش درمی‌آیند. در طی مسیر پرواز «بهترینِ شخصیِ (pbest)[18]» هر ذره، که عبارت است از بهترین مقدار تابع هدف به‌دست آمده برای آن ذره و بردار موقعیت آن، ثبت می‌شود. همچنین در هر گام، «بهترینِ کلیِ (gbest)[19]» ــ بهترین مقدار از میان بهترین‌های شخصی همه‌ی ذره‌ها و بردار موقعیت آن ــ ثبت می‌شود. در هر گام t، دریه‌ی dام از بردار سرعت یا فاصله‌ی جابه‌جایی ($\vec{v}_i(t) = (v_{i1}, v_{i2}, ..., v_{iD})$) و بردار موقعیتِ ($\vec{x}_i(t) = (x_{i1}, x_{i2}, ..., x_{iD})$) ذره‌ی iام در فضای جست‌وجو، طبق فرمول‌های بازگشتی زیر به‌هنگام می‌شود.

$$\begin{cases} v_{id}(t) = v_{id}(t-1) + c_1 r_{i1}(p_{id} - x_{id}(t-1)) + \\ \qquad c_2 r_{i2}(p_{gd} - x_{id}(t-1)) \\ x_{id}(t) = x_{id}(t-1) + v_{id}(t) \end{cases} \quad (13)$$

در این رابطه‌ها، c_1 و c_2 عددهای معمولاً ثابت‌اند و «ضریب شتاب» خوانده می‌شوند؛ r_{i1} و r_{i2} عددهای تصادفی‌اند که در بازه‌ی $[0, 1]$ با توزیع یکنواخت تولید می‌شوند؛ p_{id} دریه‌ی dام از بردار بهترینِ شخصی ذره‌ی i ($\vec{p}_i = (p_{i1}, p_{i2}, ..., p_{iD})$) و p_{gd} دریه‌ی dام از بردار بهترینِ کلیِ ($\vec{p}_g = (p_{g1}, p_{g2}, ..., p_{gD})$) است.

برای کنترل سرعت ذره‌ها در فضای جست‌وجو، از دو پارامتر $V_{\max}$ و وزن ماند (w) استفاده می‌شود.[37,38] هر دریه‌ی بردار سرعت ($v_{id}(t)$) پس از محاسبه، محدود به بازه‌ی $[-V_{\max}, V_{max}]$ می‌شود. وزن ماند (w) در فرمول ۱۳، در جمله‌ی $v_{id}(t-1)$ ضرب می‌شود. نقش w ایجاد تعادل میان جست‌وجوی کلی و جست‌وجوی محلی است. w می‌تواند یک عدد مثبت ثابت یا تابعی خطی یا غیرخطی از زمان (یا تکرار الگوریتم) باشد.[38] از نتیجه‌های تجربی چنین برمی‌آید که $w = 0.7298$ و $c_1 = c_2 = 1.49618$ هم‌گرایی خوبی به دنبال دارد.[39]

۱.۳. بهینه‌سازی دودوییِ گروه ذره‌ها

بهینه‌سازی دودوییِ گروه ذره‌ها که در سال ۱۹۹۷ معرفی شد[40] حالتی است که درآن، فضای جست‌وجو به فضای حالت‌های مختلف یک رشته‌ی بیتی اطلاق می‌شود که هر بیت آن یکی از دو حالت صفر یا ۱ را به خود می‌گیرد. رشته‌ی بیتی بردار موقعیت ذره را تشکیل می‌دهد و هر دریه‌ی بردار سرعت ذره، احتمال تغییر حالت دریه‌ی متناظر در بردار موقعیت را تعیین می‌کند. رابطه‌های به‌هنگام‌سازی سرعت و موقعیت در بهینه‌سازی دودویی گروه ذره‌ها عبارت‌اند از:

$$\begin{cases} v_{id}(t) = v_{id}(t-1) + c_1 r_{i1}(p_{id} - x_{id}(t-1)) + \\ \qquad c_2 r_{i2}(p_{gd} - x_{id}(t-1)) \\ x_{id}(t) = \begin{cases} 1 & \rho_{id} < sig\,(v_{id}(t)) \text{ اگر} \\ 0 & \text{سایر موارد} \end{cases} \end{cases} \quad (14)$$

که درآن، ρ_{id} یک عدد تصادفی است که در بازه‌ی $[0, 1]$ تولید می‌شود. تابع $sig(v_{id})$ براساس رابطه‌ی ۱۵ محاسبه می‌شود:

$$sig(v_{id}) = \frac{1}{1 + \exp(-v_{id})} \quad (15)$$

که در آن، $exp(-v_{id})$ تابع نمایی است.

۲.۳. PSO و بهینه‌سازی مسئله‌های محدودیت‌دار

شکل کلی یکی از مسائل بهینه‌سازی محدودیت‌دار (مقید) را می‌توان چنین تعریف کرد:

$$Min.\ or\ Max. f(\vec{x}),\ \ \vec{x} = (x_1, x_2, ..., x_n) \in \mathbb{R}^n \quad (16)$$

منوط به:

$$g_i(\vec{x}) \leq 0, \quad i = 1, 2, ..., p$$
$$h_j(\vec{x}) = 0, \quad j = 1, 2, ..., m$$

که در آن‌ها: $x_i^l \leq x_i \leq x_i^u,\ \ i = 1, 2, ..., n$.

در این تعریف، f، gها و hها توابع پیوسته و مشتق‌پذیرند.[41] معمول است که محدودیت‌های برابری را براساس رابطه‌ی ۱۷ به محدودیت نابرابری تبدیل کنند:

$$|h_j(\vec{x})| - \varepsilon \leq 0 \quad (17)$$

که در آن ε تولرانس مجاز (یک مقدار مثبت بسیار کوچک) است. محدودیت‌ها، فضای جست‌وجو را به دو ناحیه‌ی ممکن (F) و ناممکن تقسیم می‌کنند.

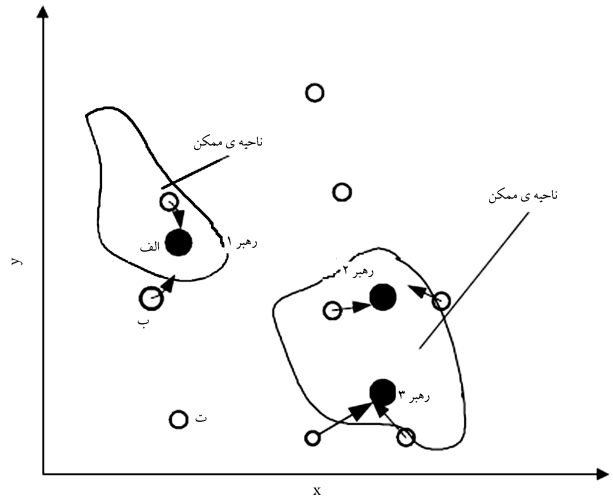
در سال ۲۰۰۷ یک الگوریتم PSO ارتقاءیافته (IPSO) معرفی شد که از یک عملگر جهش برای کاوش در ناحیه‌های مجزا در فضای جست‌وجو و پیشگیری از گیرافتادن در بهینه‌های محلی بهره می‌برد.[42] این الگوریتم هم‌چنین با جمع‌کردن برخی افراد همسایه و نزدیک به هم، به‌روشی جدید چند زیرجمعیت می‌دهد.

برای لحاظ‌کردن محدودیت‌ها ابتدا طبق رابطه‌ی ۱۷ محدودیت‌های برابری به نابرابری تبدیل می‌شوند. در اینجا نیز به‌شیوه‌ی برخی از محققین،[۴۳] در محاسبه‌ی تناسب از تابع جریمه استفاده شده که فاصله‌ی نسبی موقعیت ذره‌ها از یکدیگر و از ناحیه‌ی ممکن، یعنی درجه‌ی هم‌جواری ذره نسبت به مرز محدودیت را در نظر دارد. مقدار تابع تناسب ذره (F_{fit}) از رابطه‌ی ۱۸ محاسبه می‌شود:

$$F_{fit} = \begin{cases} f(\vec{x}), & \vec{x} \in F \\ \max\left(g_i(\vec{x}), (|h_j(\vec{x})| - \varepsilon)\,|\right) + \mathcal{N}, & \vec{x} \notin F \\ \qquad i = ۱, ۲, ..., p; \;\; j = ۱, ۲, ..., m & \end{cases} \quad (۱۸)$$

که در آن $\mathcal{N}$ یک مقدار مثبت بزرگ است. برخلاف PSO اصلی، در الگوریتم IPSO جمعیت به تعدادی زیرجمعیت تقسیم می‌شود. ذره‌ها در هر زیرجمعیت مقدار gbest (بهترین کلی) خود را از مقدار pbest (بهترینِ شخصیِ) ذره‌ی رهبر زیرجمعیت[۲۰] دریافت می‌کنند. در اینجا همسایگی‌ها و ذره‌های همسایه لزوماً ثابت نیستند. یکی از زیرجمعیت‌ها مقدار gbest را ثبت می‌کند و IPSO ذره‌یی را که در جمعیت کنونی دارای بهترین pbest باشد به‌عنوان رهبر زیرجمعیت انتخاب می‌کند.[۴۲] سپس، براساس فاصله‌ی اقلیدسی در فضای جست‌وجو، ذره‌های نزدیک به رهبر زیرجمعیت با هم زیرجمعیتی را تشکیل می‌دهند که در محاسبه‌ی موقعیت بعدی، pbest ذره‌ی رهبر را در فرمول سرعت خود به کار می‌گیرد. برای تشکیل زیرجمعیت بعدی، ابتدا زیرجمعیت قبلی از جمعیت حذف می‌شود؛ سپس با رهبر قراردادن ذره‌یی که بهترین pbest را در جمعیت باقی‌مانده دارد و تعیین همسایه‌های نزدیک به او، زیرجمعیت بعدی تشکیل می‌شود و به‌همین ترتیب همه‌ی زیرجمعیت‌ها شکل می‌گیرند (شکل ۱).

در شکل ۱ ذره‌ی الف بهترین مقدار pbest (بهترینِ شخصی) را دارد و بنابراین به‌عنوان رهبر یک زیرجمعیت انتخاب می‌شود. ذره‌ی نزدیک به ذره‌ی الف (مثلاً ذره‌ی ب) ذره‌ی الف را به‌عنوان مرکز و هسته در نظر می‌گیرد و به سوی آن پرواز می‌کند. این ذره‌ها (شامل ذره‌ی الف) زیرجمعیت ۱ را تشکیل می‌دهند که ذره‌ها را برای جست‌وجو در ناحیه‌ی ممکن سمت چپِ بالا هدایت می‌کند. رهبرهای ۲ و ۳ به‌ترتیب زیرجمعیت‌های ۲ و ۳ را برای جست‌وجو در ناحیه‌ی ممکن راست پایین رهبری می‌کنند. وقتی که تعداد زیرجمعیت‌ها به حد از پیش تعیین‌شده می‌رسد،



شکل ۱. تشکیل زیرجمعیت‌ها برای الگوریتم IPSO.[۳۷]

باقی‌مانده‌ی ذره‌ها (مثل ذره‌ی ت در شکل ۱)، از بهترین pbest کنونی در کل جمعیت (gbest) استفاده می‌کنند.[۴۲]

همچنین یک نرخ جهش پویا به PSO اضافه شده است[۴۲] که موقعیت و سرعت ذره‌ها را هم‌زمان تغییر می‌دهد. وقتی که الگوریتم اجرا می‌شود، عملگر جهش (موقعیت و سرعت) ذره‌ها را برهم می‌زند و چنانچه نرخ جهش خیلی زیاد باشد، بر جست‌وجو برای نقطه‌ی بهینه تأثیر می‌گذارد. بنابراین، باید تعداد ذره‌های جهش‌یافته کاهش یابند. نرخ جهش مورد استفاده عبارت است از:

$$temp = gen / \max_ge \quad (۱۹)$$

$$p_{mutation} = ۲(۱ - temp)^{۲} / var_num \quad (۲۰)$$

که در آن gen شماره‌ی تکرار کنونی الگوریتم، $\max_gen$ تعداد کل تکرارها و $p_{mutation}$ احتمال جهش یافتن موقعیت و سرعت ذره است. متغیر موقت $temp$ با افزایش تعداد تکرارها حین اجرای الگوریتم بزرگ‌تر می‌شود و درنتیجه $p_{mutation}$ کاهش می‌یابد تا از برهم‌خوردن ذره‌ها در تکرارهای آخر بکاهد. جهش برای ذره‌ی iام چنین بیان می‌شود:

$$v_{id} = \theta * v_{id} * ۲ * random \quad (۲۱)$$

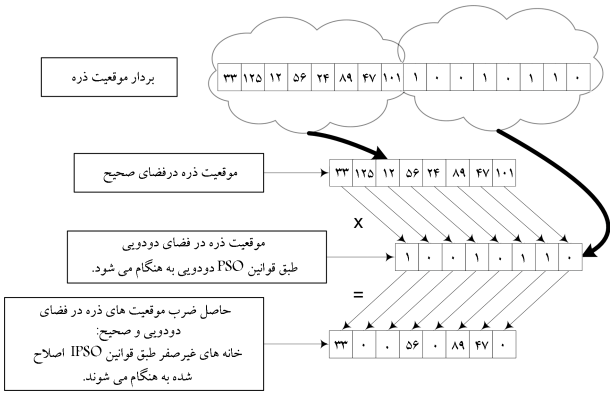
$$x_{id} = x_{id} + v_{id}, \qquad d = ۱, ۲, ..., D \quad (۲۲)$$

که در آن $\theta \in \{-۱, ۱\}$ جهتِ تولیدشده به‌طور تصادفی و $random$ عددی تصادفی در بازه‌ی $(۰, ۱)$ است. x_{id} بُعد dام موقعیت ذره‌ی i است.

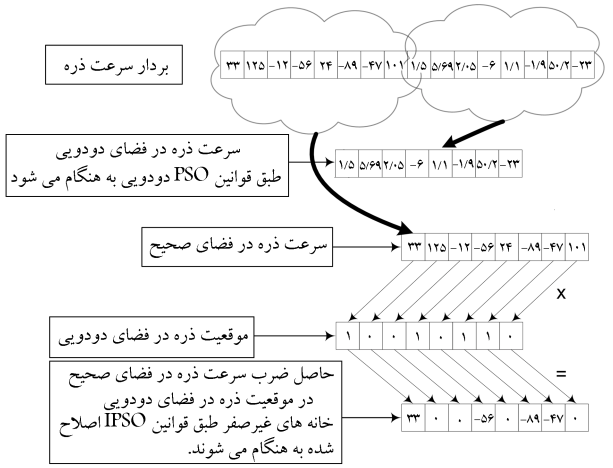
۴. الگوریتم پیشنهادی: ترکیب بهینه‌سازی گروه ذره‌های اصلاح شده و دودویی (CBIPSO) [۲۱]

الگوریتم پیشنهادی مبتنی بر دو قسمت مجزا ولی مرتبط است: قسمتی که درخصوص انتخاب M سهم از N سهمِ در دسترس عمل می‌کند و قسمتی که برای یک ترکیب M سهمیِ تعیین‌شده در هر تکرار، در فضای (عددهای) صحیح به دنبال نسبت‌های سرمایه‌گذاری بهینه است. براین اساس، هر ذره دارای دو بردار موقعیت هم‌طول است که یکی از آن‌ها بردار موقعیت ذره در فضای دودوییِ صفر و یک است (z_dها) و تصمیم‌گیری درخصوص انتخاب M سهم از N سهمِ دردسترس را نشان می‌دهد؛ بردار دیگر، بردار موقعیت ذره در فضای صحیح است که برمبنای انتخاب‌های صورت‌گرفته در بردار موقعیت دودویی (درایه‌های متناظری که مقدار ۱ می‌گیرند) به‌هنگام می‌شود (x_dها). این بردار، نسبت‌های سرمایه‌گذاری را تعیین می‌کند. در این الگوریتم، دو بردار موقعیت گفته‌شده، درکنار هم به تشکیل بردار سطری موقعیت ذره می‌انجامند (شکل ۲).

همچنین بردار سطریِ سرعت ذره نیز از دو قسمت تشکیل می‌شود (شکل ۳). قسمت اول تعیین‌کننده‌ی سرعت ذره در فضای تصمیم‌گیری دودویی است و شامل عددهای حقیقی در بازه‌ی $(-V_{\max}B_d, V_{\max}B_d)$ است. عبارت $V_{\max}B_d$ نشان‌گر پارامتر $V_{\max}$ برای بُعد (سهم) dام در قسمت دودویی است. قسمت دوم، سرعت ذره را در فضای صحیح نشان می‌دهد و درایه‌های آن از عددهای صحیحی در بازه‌ی $(-V_{\max}I_d, V_{\max}I_d)$ تشکیل می‌شود. $V_{max}I_d$، پارامتر V_{max} برای بُعد (سهم) dام در قسمت صحیح است. به‌هنگام‌سازی دو قسمت صحیح و دودویی، جداگانه و به‌ترتیب طبق روش‌های IPSO اصلاح شده و بهینه‌سازی دودویی گروه ذره‌ها، که پیش‌تر توضیح داده شد، صورت می‌پذیرد.



شکل ۲. بردار موقعیت ذره در الگوریتم CBIPSO و اجزای آن. در این شکل، تعداد کل سهم‌های در دسترس (N) ۸ و اندازه‌ی سبد (M) ۴ است.



شکل ۳. بردار سرعت ذره در الگوریتم CBIPSO و اجزای آن.

برای قسمت صحیح، موقعیت ذره‌ها با عددهای صحیح تصادفی بین حد پایین ($lbd_d = \frac{B_{Low_d}}{c_d}$) و حد بالا ($ubd_d = \frac{B_{Up_d}}{c_d}$) مقداردهی اولیه می‌شوند. سرعت نیز با عددهای تصادفی صحیح در بازه‌ی $(-V_{max}I_d, V_{max}I_d)$ مقداردهی اولیه می‌شود. از قرار دادن $V_{\max} = X_{max}$ در هر بُعد، نتایج بهتری به دست می‌آید.[۴۴] $X_{\max}$ کران بالای فضای جست‌وجو در هر بُعد است، و بنابراین می‌تواند برای هر بعد عددی متفاوت باشد. $V_{\max}I$، به‌صورت یک بردار Nتایی درنظرگرفته می‌شود که مقدار هر درایه‌ی d آن ضریبی از $(ubd_d - lbd_d)$ است.

برای قسمت دودویی، هر بردار سطری موقعیت از جایگشت تصادفی M عددِ ۱ و $N - M$ عددِ صفر تشکیل می‌شود. درایه‌های بردارهای اولیه‌ی سرعت عددهایی حقیقی‌اند که به‌طور تصادفی از بازه‌ی $[-۱, ۱]$ انتخاب می‌شوند.

برای افزایش سرعت الگوریتم، مقدار بهترینِ کلی و بهترین‌های شخصی اولیه‌ی ذره‌ها با یک عدد نسبتاً بزرگ برابر می‌شود. این عدد بزرگ ($\mathcal{N}$)، در اینجا برابر با مقدار کل بودجه‌ی قابل تخصیص (B) است. وقتی که مقدار بهترین‌های شخصی اولیه عدد بزرگی باشد، احتمال تغییر آن‌ها در نخستین محاسبه‌ی تناسب برای هر ذره بالا می‌رود. درنتیجه، ذره‌هایی که فاصله‌ی نزدیک‌تری به ناحیه‌ی ممکن دارند شانس بیشتری برای رهبری زیرجمعیت‌ها خواهند داشت. اگر بهترینِ کلیِ اولیه نیز زیاد باشد، پس از محاسبه‌ی تناسب ذره‌ها و به‌هنگام کردن بهترین‌های شخصی آن‌ها، فقط ذره‌هایی که محدودیت‌ها را ارضا کنند حق تعیین بهترین کلی را خواهند داشت.

در حلقه‌ی به‌هنگام‌سازی سرعت و موقعیت ذره‌ها، شرط توقف این است که تعداد موقعیت‌های ممکن بیشینه شود یا تعداد تکرار از ده برابر بیشینه موقعیت‌های ممکن بگذرد. مقدار تناسب برای هر ذره محاسبه می‌شود. اگر ذره در ناحیه‌ی ممکن باشد، تناسب ذره طبق رابطه‌ی ۲۳ محاسبه می‌شود:

$$F_{fit} = f(\vec{X^*}) + ۰/۹ * (۱ - \frac{\sum_{d=۱}^{N} x_d z_d c_d}{B}) \qquad (۲۳)$$

که در آن:

$$\vec{X^*} = \vec{x} \otimes \vec{z} = (x_۱ z_۱, ..., x_d z_d) \qquad (۲۴)$$

$\vec{X^*}$ حاصل‌ضرب نقطه به نقطه‌ی بردار $\vec{x}$ (قسمت صحیح موقعیت ذره) و بردار $\vec{z}$ (قسمت دودویی موقعیت ذره) است. رابطه‌ی ۲۳ در مقایسه با رابطه‌ی ۱۸ دارای یک قسمت اضافی است. در اینجا، این قسمت به‌اقتضای مسئله‌به فرمول تناسب اضافه شده است تا ضمن کمینه‌شدن تابع هدف، رابطه‌ی ۹ (محدودیت بودجه) نیز به سمت تساوی بگراید. همراه با محاسبه‌ی تناسب، شمارنده‌ی تعداد موقعیت‌های ممکن یکی افزایش می‌یابد.

اگر موقعیت ذره در ناحیه‌ی ممکن نباشد، براساس رابطه‌ی ۱۸ تناسب ذره مطابق زیر تعیین می‌شود. برای تعیین میزان انحراف از محدودیت ۱۱ داریم:

$$\Delta^-_{ss'} = \begin{cases} \sum_{d_{s'}} w_{d_{s'}} - \sum_{d_s} w_{d_{s'}}, \sum_{d_{s'}} w_{d_{s'}} > \sum\sum_{d_s} w_{d_s} \\ ۰ \qquad o.w. \end{cases}$$

$$y_s, y_{s'} \neq ۰, s' > s \qquad (۲۵)$$

که در آن، s' شماره‌ی اولین بخش بعد از بخش s است که برای آن $y_{s'} \neq ۰$. مقدار $\Delta^-_{ss'}$ از رابطه‌ی ۱۲ به دست می‌آید اما، برای هر s و s' متوالی که $y_s, y_{s'} \neq ۰$ داریم:

$$\Delta^-_s = \sum_{s,s'} \Delta^-_{s,s'} \qquad (۲۶)$$

به دست آوردن میزان عدول از سایر محدودیت‌ها کار آسانی است؛ بنابراین، تناسب برابر است با:

$$F_{fit} = \max \Bigg[(B - \sum_{d=۱}^{N} x_d z_d c_d), (BR - \sum_{d=۱}^{N} x_d z_d c_d \bar{R}_d), \left| \sum_{d=۱}^{N} z_d - M \right|, \Delta^-_s \Bigg] + \mathcal{N} \qquad (۲۷)$$

که در آن $\mathcal{N}$ یک عدد بزرگ است و سایر متغیرها مطابق رابطه‌های ۷ تا ۹ هستند.

پس از محاسبه‌ی تناسب، بهترین‌های شخصی ذره‌ها در مقایسه با تناسب‌شان به‌هنگام می‌شوند؛ به این صورت که اگر مقدار تناسب ذره‌یی از بهترینِ شخصی‌اش کم‌تر باشد، مقدار تناسب جایگزین بهترینِ شخصی می‌شود. در این صورت، موقعیت کنونی ذره نیز به‌عنوان موقعیت بهترینِ شخصی ثبت می‌شود.

کم‌ترین مقدار بین مقدارهای تناسب ذره‌ها و بهترین کلیِ کنونی، به‌عنوان بهترینِ کلی جدید ثبت می‌شود. سپس، اگر بهترینِ کلی با کم‌ترین مقدار تناسب‌ها برابر باشد، موقعیت بهترین کلی با موقعیت ذره‌یی که کم‌ترین تناسب را دارد برابر می‌شود.

در هر تکرار، تعداد رهبران زیرجمعیت‌ها به‌طور پویا براساس تعداد ذره‌های واقع در ناحیه‌ی ممکن تعیین می‌شود. اگر هیچ ذره‌ی ممکنی در آن تکرار وجود نداشته

باشد، تعداد رهبران زیرجمعیت‌ها برابر با ۱ منظور می‌شود. از سوی دیگر، سقفی برای این تعداد لحاظ می‌شود (مثلاً ۱۰)؛ اگر تعداد ذره‌های ممکن از این سقف بالاتر باشد، تعداد رهبران با مقدار سقف مساوی می‌شود. تعداد رهبران که برابر با بیشترین تعداد زیرجمعیت‌هاست براساس فرمول ۲۸ به دست می‌آید و مختص این تحقیق است:

$$Max\,No\,of\,Sub\,Pops = \begin{cases} \min(No\,of\,feasible particles,\ cte) & \exists \vec{x}_i \in F \\ ۱ & \nexists \vec{x}_i \in F \end{cases} \quad (۲۸)$$

که در آن، F ناحیه‌ی ممکن است. در این تحقیق، ثابت cte برابر با ۱۰ فرض می‌شود.

روشی که در این نوشتار برای تشکیل زیرجمعیت‌ها در پیش گرفته شده است با روش‌های پیشین[۴۲] تفاوت دارد. در این روش، ابتدا ذره‌ها براساس بهترینِ شخصی‌شان ــ از بهترین تا بدترین ــ مرتب می‌شوند. سپس از بهترین ذره شروع کرده و به تعداد معلوم ذره‌های رهبر تعیین و از جمعیت جدا می‌شوند. آنگاه، از میان باقی‌مانده‌ی جمعیت و براساس معیارهای فاصله‌ی اقلیدسی و فاصله‌ی همینگ برای هر رهبر عضوگیری و زیرجمعیت‌ها تشکیل می‌شوند. روش پیشین[۴۲] باعث می‌شود که اگر چند ذره‌ی نزدیک به هم ــ در مقایسه با ذره‌های دیگر ــ مقدار تناسب خیلی خوبی در جمعیت داشته باشند، همگی اعضای یک زیرجمعیت شوند و زیرجمعیت‌های دیگر رهبری ضعیفی داشته باشند، و عملاً تأثیر جست‌وجوی دیگر زیرجمعیت‌ها در سرعت دست‌یابی به پاسخ بهینه به کم‌ترین حد برسد. اما روش جدید می‌تواند این نقیصه را برطرف سازد. مقایسه‌ی عملی این دو روش با هم نشان می‌دهد که روش جدید عملکرد و نتیجه‌ی بهتری داشته است.

فاصله‌ی اقلیدسی ذره‌ها از رهبر در فضای جست‌وجوی صحیح، با یک شعاع همسایگی سنجیده می‌شود. ذره‌هایی که درون شعاع همسایگیِ رهبر قرار می‌گیرند، با او تشکیل زیرجمعیت می‌دهند. علاوه بر فاصله‌ی اقلیدسی، معیار دیگری نیز برای عضویت در زیرجمعیتِ یک رهبر قرار داده شده است و آن فاصله‌ی همینگ موقعیت دودویی ذره با موقعیت دودویی رهبر است که باید از یک مقدار معین کم‌تر باشد. ذره‌های هر زیرجمعیت، ازجمله رهبر آن، از موقعیت بهترینِ شخصیِ رهبر به‌عنوان بهترینِ کلی محلی در فرمول به‌هنگام‌سازی سرعت استفاده می‌کنند. وزن ماند (w) در هرتکرار به‌طور نزولی تغییر می‌کند. روش محاسبه‌ی وزن ماند چنین است:

$$w = w_{\max} - \frac{No\,of\,Feasible\,Positions}{Max\,Feasible\,Positions} \times (w_{\max} - w_{\min}) \quad (۲۹)$$

که در آن، $No\,of\,Feasible\,Positions$ شمارنده‌ی تعداد موقعیت‌های ممکن تاکنون و $Max\,Feasible\,Positions$ بیشینه موقعیت‌های ممکن است. چون در هر تکرار، ممکن است تعداد موقعیت‌های احتمالی متفاوت باشد، طبق رابطه‌ی ۲۹ وزن ماند به‌طور خطی کاهش نمی‌یابد. دلیل این که در رابطه‌ی ۲۹ تعداد موقعیت‌های ممکن به‌جای تعداد تکرار به کار برده شده این است که تعداد موقعیت‌های ممکن معیاری برای حرکت مؤثر به سمت بهینه‌سازی است، در حالی که موقعیت‌های ناممکن یا به‌عبارتی موقعیت‌هایی که در محدودیت‌ها صدق نمی‌کنند عملاً نقش مهمی در پیشبرد مسئله‌ندارند. در تکراری که فاقد هرگونه موقعیت ممکنی است نباید وزن ماند تغییر کند. به‌همین دلیل است که شرط پایان حلقه‌ی تکرار نیز رسیدن تعداد موقعیت‌های ممکن به حد معین قرار داده شده است. در اینجا، $W_{\max} = ۱/۰۵$ و $W_{\min} = ۰/۴$ انتخاب شده‌اند.[۳۸] مشابه همین شیوه برای تعیین نرخ جهش درپیش گرفته شده است:

$$p_{mutation} = ۰/۲۵ \left(۱ - \frac{No\,of\,Feasible\,Positions}{Max\,Feasible\,Positions}\right)^{۲} \quad (۳۰)$$

در اینجا نیز اگرچه نرخ جهش با اجرای الگوریتم کاهش می‌یابد، معذلک این کاهش خطی نیست. این موضوع وجه تمایز دیگری در روش پیشنهادی در مقایسه با روش پیشین می‌باشد. (رابطه‌ی ۲۰ را ملاحظه فرمایید).

در به‌هنگام‌سازی سرعت برای قسمت دودویی و صحیح، مقدار ضریب‌های شتاب $c_۱$ و $c_۲$ براساس بخش ۳ معادل ۱/۴۹۶ فرض شده‌اند.

نخست قسمت دودویی سرعت و موقعیت ذره‌ها به‌هنگام می‌شود. به‌هنگام‌سازی براساس رابطه‌ی ۱۴ انجام می‌شود با این تفاوت که به‌جای استفاده از موقعیت بهترین کلی، از موقعیت بهترین شخصیِ رهبرِ زیرجمعیت استفاده می‌شود، یعنی p_{ld} جایگزین p_{gd} می‌شود. $r_۱$ و $r_۲$ برای هر ذره و هر بُعد، از بازه‌ی بسته‌ی [۰, ۱] به‌طور تصادفی مقداردهی می‌شوند. پس از به‌هنگام‌سازی سرعت، سرعت جدید در هر بُعد با حد بالای سرعت $V_{\max}B_d$ مقایسه می‌شود تا فراتر از آن قرار نگیرد. سپس، به‌هنگام‌سازی موقعیت صورت می‌گیرد.

برای هر ذره اگر همه‌ی شرط‌های زیر برقرار باشند، با احتمالی که نرخ جهش تعیین می‌کند، جهش روی موقعیت دودویی اعمال می‌شود.

- اندازه‌ی سبد کم‌تر از سهم‌های در دسترس باشد ($M < N$) و درنتیجه: $\exists d \in \{۱, ۲, ..., N\}; z_d = ۰$؛
- دست‌کم یک بیت (بُعد) از موقعیت دودویی ذره ۱ باشد؛
- تعداد بیت‌های (بُعدهای) ۱ موقعیت دودویی ذره کوچک‌تر یا مساوی اندازه‌ی سبد (M) باشد.

برای جهش‌دادن قسمت دودویی ابتدا بُعدی از موقعیت دودویی به‌طور تصادفی انتخاب می‌شود و اگر مقدار آن ۱ بود به صفر تغییر داده می‌شود. اگر مقدار بیت ۰ بود مجدداً بیت دیگری تصادفاً انتخاب می‌شود و این روند تا آنجا ادامه می‌یابد که یک بیت ۱ پیدا شود. پس از این مرحله، در میان بقیه‌ی بیت‌ها جست‌وجوی تصادفی برای یافتن بیتی که ۰ باشد انجام می‌شود تا مقدار آن به ۱ تبدیل شود.

برای هر ذره، به‌هنگام‌سازی قسمت صحیح در بُعدهایی انجام می‌شود که مقدار بیت (بُعد) دودویی نظیر آن‌ها در قسمت دودویی برابر با ۱ باشد. به‌عبارت دیگر، نسبت سرمایه‌گذاری سهمی تغییر داده می‌شود که برای سبد انتخاب شده باشد. این کار ضمن آن که منطقی به نظر می‌رسد، مقدار قابل توجهی از حجم محاسبات می‌کاهد. ضریب‌های شتاب به‌طور تصادفی از بازه‌ی [۰/۵, ۱/۵] انتخاب می‌شوند.[۲۵] با توجه به آن که حاصل ضرب $x_d c_d$ تعیین‌کننده‌ی نسبت دارایی dام در سبد است (رابطه‌ی ۶) و نیز c_dها با هم متفاوت‌اند، تأثیر ناشی از یک واحد تغییر x_d در w_d با تأثیر ناشی از یک واحد تغییر x_j در w_j (که $d \neq j$) متفاوت است. از این رو برای نرمال‌سازی تأثیر جست‌وجو در بُعدهای مختلف فضا از یک ضریب نرمال‌سازی برای هر بُعد از موقعیت قسمت صحیح استفاده می‌کنیم که از رابطه‌ی ۳۱ به دست می‌آید.

$$n_d = (\max\{c_d | d = ۱, ۲, ..., N\})/c_d) \quad (۳۱)$$

در به‌هنگام‌سازی سرعت، این ضریب در وزن ماند ضرب می‌شود. بنابراین، فرمول

بردار سرعت برای ذره‌ی iام و بُعد (سهم) dام، به رابطه‌ی ۳۲ تبدیل می‌شود:

$$v_{id}(t) = n_d w v_{id}(t-1) + c_1 r_{id1}\left(p_{id} - x_{id}(t-1)\right) + c_2 r_{id2}\left(p_{ld} - x_{id}(t-1)\right) \quad (32)$$

هر درایه‌ی بردار سرعت پس از محاسبه گرد می‌شود. r_1 و r_2 برای هر ذره و هر بُعد از رابطه‌ی ۳۳ معادل $rand + 0/5$ به‌دست می‌آید[۴۳] که در آن، عددی تصادفی در بازه‌ی $[0, 1]$ است. پس از این، اگر سرعت از $V_{\max} l_d$ بیشتر یا از $-V_{\max} l_d$ کم‌تر باشد مقدار سرعت به سرعت‌های یادشده بازنشانده می‌شود. موقعیت نیز طبق رابطه‌ی ۱۳ به‌هنگام می‌شود. سپس بُعدهای موقعیت با حدهای پایین و بالا (lbd_d و ubd_d) مقایسه می‌شوند. چنانچه بُعدی از موقعیت خارج از حدهای گفته شده قرار گیرد با کف یا سقف آن حد برابر می‌شود.

پس از این مرحله موقعیت‌ها براساس نرخ جهش، جهانده می‌شوند. رابطه‌های ۲۱ و ۲۲ ملاک اعمال جهش هستند. پس از اعمال جهش، بار دیگر بُعدهای موقعیت با حدهای پایین و بالا (lbd_d و ubd_d) مقایسه و در صورت نیاز اصلاح می‌شوند. به این ترتیب برقراری محدودیت ۱۰ تضمین می‌شود. مراحل الگوریتم CBIPSO پیشنهادی عبارت است از:

۱. سرعت و موقعیت اولیه‌ی ذره‌ها را ایجاد کن؛

۲. مقدار تابع هدف را برای هر ذره محاسبه کن؛

۳. بهترین شخصی هر ذره را برابر با جمع تابع هدف محاسبه‌شده برای آن ذره و یک عدد بزرگ ($\mathcal{N}$) قرار بده. بهترین کلی را برابر با جمع بیشترین مقدار تابع هدف محاسبه‌شده برای ذره‌ها و $\mathcal{N}$ قرار بده؛

۴. بردار موقعیت بهترین شخصی هر ذره را برابر با موقعیت اولیه قرار بده؛

۵. ضریب‌های نرمال‌سازی حرکت در فضای صحیح را طبق رابطه‌ی ۳۱ حساب کن؛

۶. مقدار تابع هدف را برای هر ذره محاسبه کن؛

۷. برای هر ذره اگر در ناحیه‌ی ممکن قرار دارد، یعنی در همه‌ی رابطه‌های ۷ تا ۱۲ صدق می‌کند، تناسب را از رابطه‌ی ۲۳ محاسبه کن و شمارنده‌ی تعداد موقعیت‌های ممکن را یکی افزایش بده؛

۸. اگر ذره در ناحیه‌ی ناممکن قرار دارد، یعنی دست‌کم یکی از شرط‌های ۷ تا ۱۲ را برآورده نمی‌کند، تناسب ذره را از رابطه‌ی ۲۷ محاسبه کن؛

۹. بهترین‌های شخصی ذره‌ها و بردار موقعیت آن‌ها را حساب کن؛

۱۰. بهترین کلی و بردار موقعیت بهترین کلی را به‌دست بیاور؛

۱۱. بیشترین تعداد زیرجمعیت‌ها، و به‌عبارت دیگر تعداد رهبران زیرجمعیت‌ها را از رابطه‌ی ۲۸ تعیین کن؛

۱۲. به تعداد تعیین شده در مرحله‌ی ۱۱، رهبران را که به‌ترتیب دارای کم‌ترین pbest هستند، مشخص و از جمعیت حذف کن؛

۱۳. زیرجمعیت‌ها را تشکیل بده؛

۱۴. وزن ماند نزولی را طبق فرمول ۲۹ حساب کن؛

۱۵. نرخ جهش را براساس رابطه‌ی ۳۰ تعیین کن؛

۱۶. سرعت و موقعیت دودویی را برای هر ذره به‌هنگام کن (طبق ۱۴)؛

۱۷. جهش را با احتمال نرخ جهش مرحله‌ی ۱۵، روی موقعیت دودویی اعمال کن؛

۱۸. سرعت صحیح را براساس رابطه‌ی ۳۲ و موقعیت صحیح را طبق ۱۳ حساب کن؛

۱۹. جهش را با احتمال نرخ جهش مرحله‌ی ۱۵، روی موقعیت صحیح اعمال کن؛

۲۰. شمارنده‌ی تعداد تکرار را یکی افزایش بده؛

۲۱. اگر شمارنده‌ی تعداد موقعیت‌های ممکن کم‌تر از بیشینه‌ی تعیین شده است، و تعداد تکرار کم‌تر از ده‌برابر بیشینه تعداد موقعیت‌های ممکن است، به مرحله‌ی ۶ برو؛

۲۲. بهترین کلی را به‌عنوان کم‌ترین ریسک به‌دست آمده و موقعیت بهترین کلی را به‌عنوان جواب مسئله‌نمایش بده.

۵. نتایج تجربی

۱.۵. سنجش اعتبار و کارایی الگوریتم CBIPSO

الگوریتم CBIPSO و الگوریتم ژنتیک،[۲۴] برای سه مسئله‌ی نمونه‌ی ۹، ۳۰ و ۱۵۰ سهمی اجرا شده‌اند. الگوریتم‌ها در قالب کدهای MATLAB®، روی رایانه‌یی با ویژگی‌های Intel® Pentium® ۴ ۳/۰GHz, ۱۰۲۴KB DDRII RAM, FSB ۵۳۳MHz به اجرا درآمده‌اند. برای هر مسئله‌و هر تنظیم خاص از پارامترها، الگوریتم‌ها ۲۰ بار اجرا شده‌اند. سپس نتایج آماریِ جواب‌های (ریسک‌های) حاصل از ۲۰ اجرا ثبت و مقایسه شده‌اند. برای الگوریتم ژنتیک مطابق تنظیم‌های ثبت‌شده از دیگر مطالعات[۲۴] تعداد تکرار ۱۰۰۰۰، اندازه‌ی جمعیت ۲۰ و نرخ جهش ۰/۵ تعیین شده‌اند. در الگوریتم CBIPSO تعداد موقعیت‌های ممکن برابر با ۳۰۰۰ قرار داده شده است. همچنین نرخ جهش کاهشی از ۰/۵ تا ۰/۲۵ تغییر می‌کند. اندازه‌ی جمعیت نیز برابر با ۲۰ قرار داده شده است.

نتایج حاصل از اجرای دو الگوریتم با استفاده از چند معیار مقایسه شده‌اند. بهترین (کم‌ترین) ریسک در میان ریسک‌های حاصل از چند بار اجرای الگوریتم برای حل مسئله، میانگین ریسک‌ها، انحراف معیار ریسک‌ها و میانگین زمان اجرای الگوریتم چهار معیار مقایسه دو روش بوده‌اند.

با مقادیر تنظیم‌شده‌ی فوق برای پارامترها، نتایج به‌دست آمده برای مسئله‌ی ۹ سهمی مطابق جدول‌های ۱ و ۲ هستند. نتیجه‌ی مقایسه‌ی این دو جدول را در جدول ۳ مشاهده می‌کنید. در این جدول، خانه‌هایی که در آنها کلمه‌ی «بهتر» مشاهده می‌شود، نشان‌دهنده‌ی این است که در شرایط آن خانه و براساس معیار سمت راست آن، الگوریتم CBIPSO بهتر از الگوریتم ژنتیک عمل کرده است. برعکس، در خانه‌هایی که کلمه‌ی «بدتر» دیده می‌شود عملکرد الگوریتم ژنتیک،[۲۴] بهتر از CBIPSO بوده است. با نگاهی کلی به این جدول به‌راحتی می‌توان دریافت که عملکرد کلی CBIPSO در این مسئله‌بهتر از الگوریتم ژنتیک است. نمودار شکل ۴ میانگین ریسک‌ها و شکل ۵ زمان میانگین اجرا را برای هر دو الگوریتم نشان می‌دهد.

در الگوریتم ژنتیک،[۲۴] مجموع نسبت‌های سرمایه‌گذاری همه‌ی جواب‌ها دقیقاً برابر با ۱ است. این امر البته در CBIPSO برقرار نیست. در CBIPSOتعداد اجراهایی که در نتیجه‌ی آن‌ها بودجه به‌طورکامل به سرمایه‌گذاری اختصاص می‌یابد (جمع نسبت‌های سرمایه‌گذاری برابر با ۱ می‌شود) کم است، اما تجربه نشان می‌دهد که در همه‌ی موارد بیش از ۹۷٪ سرمایه به امر سرمایه‌گذاری اختصاص یافته است.

جدول ۱ * نتایج اجرای CBIPSO برای مسئله‌ی ۹ سهمی: رابطه‌ی اندازه‌ی سبد و نرخ بهره‌ی مورد انتظار.

تعداد سهم‌ها در سبد	نرخ بهره‌ی مورد انتظار (٪) <<	۷	۸	۹	۱۰	میانگین
۴	بهترین (کم‌ترین) واریانس	$۸٫۴۹ \times ۱۰^{-۹}$	$۸٫۴۹ \times ۱۰^{-۹}$	$۵٫۲۳ \times ۱۰^{-۶}$	$۱٫۴۹ \times ۱۰^{-۵}$	$۷٫۳۳ \times ۱۰^{-۶}$
	میانگین واریانس‌ها	$۹٫۲۶ \times ۱۰^{-۵}$	۰٫۰۰۱۸۴۳	$۸٫۳۰ \times ۱۰^{-۵}$	۰٫۰۱۶۰۸۱	۰٫۰۳۵۱۶۰
	انحراف معیار واریانس‌ها	۰٫۰۰۰۱۰۵	۰٫۰۰۵۳۶۳	$۷٫۹۶ \times ۱۰^{-۵}$	۰٫۰۳۸۶۰۳	۰٫۰۸۸۰۱۴
	**	۱۸	۲۰	۱۹	۲۰	۱۹٫۶۲۵
	زمان میانگین (ثانیه)	۱۰٫۶۶	۱۳٫۹۶	۱۸٫۷۴	۳۱٫۹۰	۳۲٫۷۷
	تعداد تکرار متوسط	۱۷۳۷	۲۰۵۳	۲۵۷۹	۴۳۷۶	۴۷۵۰

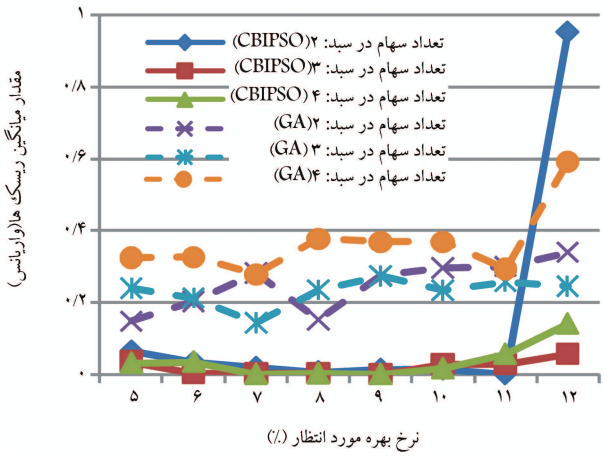
جدول ۲ * نتایج اجرای الگوریتم ژنتیک[۲۴] برای مسئله‌ی ۹ سهمی: رابطه‌ی اندازه‌ی سبد و نرخ بهره‌ی مورد انتظار.

تعداد سهم‌ها در سبد	نرخ بهره‌ی مورد انتظار (٪) <<	۷	۸	۹	۱۰	میانگین
۴	بهترین (کم‌ترین) واریانس	۰٫۰۵۸۵۷۷	۰٫۰۴۰۱۹۷	۰٫۰۰۳۷۵۴	۰٫۰۱۰۶۴۷	۰٫۰۲۱۲۷۴
	میانگین واریانس‌ها	۰٫۲۷۸۶۲۷	۰٫۳۷۷۴۹۷	۰٫۳۶۸۵۱۲	۰٫۳۶۸۷۰۴	۰٫۳۶۶۱۳۸
	انحراف معیار واریانس‌ها	۰٫۲۱۵۴۳۲	۰٫۲۹۴۳۱۹	۰٫۳۰۰۴۷۴	۰٫۳۷۴۰۲۳	۰٫۲۹۳۲۳۹
	**	۲۰	۲۰	۲۰	۲۰	۲۰
	زمان میانگین (ثانیه)	۳۴٫۴۳	۴۷٫۳۷	۷۶٫۶۸	۱۵۰٫۱۸	۳۶۲٫۵۴
	تعداد تکرار متوسط	۱۰۰۰	۱۰۰۰	۱۰۰۰	۱۰۰۰	۱۰۰۰

جدول ۳ * مقایسه‌ی نتایج اجرای دو الگوریتم برای مسئله‌ی ۹ سهمی (مقایسه‌ی جدول‌های ۱ و ۲).

تعداد سهم‌ها در سبد	نرخ بهره‌ی مورد انتظار (٪) <<	۷	۸	۹	۱۰	میانگین
۴	بهترین (کم‌ترین) واریانس	بهتر	بهتر	بهتر	بهتر	بهتر
	میانگین واریانس‌ها	بهتر	بهتر	بهتر	بهتر	بهتر
	انحراف معیار واریانس‌ها	بهتر	بهتر	بهتر	بهتر	بهتر
	**	بدتر	برابر	بدتر	برابر	بدتر
	زمان میانگین (ثانیه)	بهتر	بهتر	بهتر	بهتر	بهتر

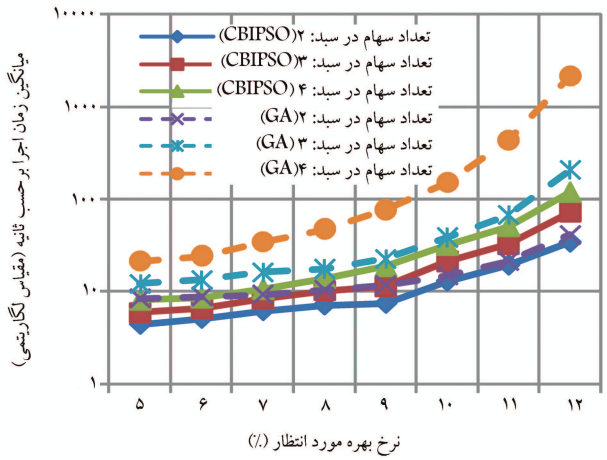
* اطلاعات جداول ۱ تا ۳ مختصر شده‌اند. جداول کامل‌تر را می‌توان در منبع[۴۶] مشاهده کرد. همچنین اطلاعات فوق بر مبنای ۲۰ اجرا تهیه شده است و بودجه تخصیص یافته در جواب ها بیش از ۹۹٫۸٪ می باشد.



شکل ۴. میانگین ریسک‌ها برای مسئله‌ی ۹ سهمی؛ خط‌های پر مربوط به الگوریتم CBIPSO و خط‌چین‌ها مربوط به الگوریتم ژنتیک‌اند.

در این تحقیق، با نگاهی سخت‌گیرانه‌تر، تعداد مواردی که در ۲۰ اجرا بیش از ۹۹٫۸٪ سرمایه را تخصیص دهند شمارش شده‌اند. نکته‌ی مهم در جدول ۳ این است که تعداد تکرار، معیاری برای مقایسه‌ی دو الگوریتم قرار نگرفته است و این طبیعی است، زیرا تعداد تکرار در ارتباط با زمان اجرا اهمیت می‌یابد که در قسمت «زمان میانگین» این مقایسه صورت گرفته است. از طرفی، با توجه به تفاوت‌های ماهوی دو الگوریتم، تعداد تکرار در آن‌ها دارای مفهوم یکسان نیست.

با بررسی نتایج در جدول‌های ۱ و ۲ درمی‌یابیم که بهترین ریسکِ به‌دست آمده به‌وسیله‌ی CBIPSO از مرتبه‌ی $۱۰^{-۶}$ است. در حالی که بهترین ریسک‌ها در الگوریتم ژنتیک از مرتبه‌ی $۱۰^{-۳}$ و $۱۰^{-۲}$ هستند. بنابراین، CBIPSO قادر است به پاسخ‌های بسیار بهتری دست یابد. همچنین از لحاظ «میانگین ریسک‌ها» نیز (به‌جز یک مورد) عملکرد CBIPSO به‌طرز چشم‌گیری بهتر بوده است. این امر در نمودار شکل ۴ به‌روشنی پیداست. به‌طورکلی (با صرف‌نظرکردن از یک مورد استثنا) انحراف معیار ریسک‌های CBIPSO کم‌تر از الگوریتم ژنتیک است که حاکی از ثبات[۲۲] و تکرارپذیری بیشتر در جواب‌های به‌دست آمده از روش CBIPSO است.



شکل ۵. میانگین زمان برای مسئله‌ی ۹ سهمی؛ خط‌های پر مربوط به الگوریتم CBIPSO و خط‌چین‌ها مربوط به الگوریتم ژنتیک‌اند. برای تفکیک بین خط‌ها در محور عمودی از مقیاس لگاریتمی استفاده شده است.

«زمان میانگین» دست‌یابی به جواب نیز یکی از دو عامل مهمی است که در کنار عامل «میانگین ریسک‌ها» معیار اصلی سنجش الگوریتم‌ها را تشکیل می‌دهد. به‌خوبی پیداست که CBIPSO در مدت‌زمان کم‌تری می‌تواند جواب‌های بهتری را ارائه کند.

مشابه جدول ۳، مقایسه‌یی بین دو الگوریتم برای مسئله‌ی ۱۵۰ سهمی[۴۶] صورت پذیرفته است. در آزمایش‌های صورت‌گرفته مشخص شد که الگوریتم ژنتیک گاهی در عمل نمی‌تواند اجرا را به پایان برساند و به‌اصطلاح «گیر» می‌کند. گمان می‌رود زمان ماندن در این وضعیت بی‌نهایت باشد، زیرا دست‌کم برای ۱۰ ساعت تحت آزمایش قرارگرفته است. در اینجا این وضعیت، «وضعیت گیر» الگوریتم ژنتیک نامیده می‌شود. احتمال وقوع وضعیت گیر در برخی از حالت‌ها افزایش می‌یابد به‌گونه‌یی که با چندین‌بار سعی مجدد نیز الگوریتم ژنتیک قادر به تکمیل ۲۰ اجرای متوالی نیست. در این حالت‌ها، البته الگوریتم CBIPSO قادر به ارائه‌ی جواب بوده است.

برای مسئله‌ی ۳۰ سهمی و با شرط ۲۰ اجرای متوالی، الگوریتم ژنتیک برای تمام حالت‌ها در وضعیت گیر قرارگرفته و بنابراین، امکان دست‌یافتن به نتایج مناسب برای مقایسه‌ی دو الگوریتم در زمان معقول فراهم نشد. سایر تنظیم‌ها از قبیل نرخ جهش، اندازه‌ی جمعیت و تعداد تکرار مقداردهی شده‌اند.[۲۴]

مقایسه‌ی دو الگوریتم براساس معیارهای مختلف در جدول ۴ خلاصه شده است. چنان که مشاهده می‌شود CBIPSO فقط در تخصیص کل بودجه به سرمایه‌گذاری بدتر از الگوریتم ژنتیک عمل می‌کند. در جدول ۱ چند مورد به چشم می‌خورد که طی آنها CBIPSO توانسته است همانند الگوریتم ژنتیک در ۲۰ اجرای متوالی همه‌ی بودجه را به سرمایه‌گذاری اختصاص دهد. بنابراین، می‌توان انتظار داشت که اگر پارامترهای CBIPSO به تناسبِ مسئله‌مقداردهی شوند، این نقیصه‌ی CBIPSO نیز برطرف شود.

۲.۵. تحلیل حساسیت و تنظیم پارامترها

یافتن تنظیم بهینه‌ی پارامترها و متغیرهای الگوریتم و مسئله، خود مسئله‌ی بهینه‌سازی دیگری است. در این تحقیق با دو دسته پارامتر مواجه هستیم: پارامترهای الگوریتم و پارامترهای مسئله. پارامترهای الگوریتم عبارت‌اند از پارامترهایی که جدا از مسئله و دستاورد فلسفه‌ی نهفته در الگوریتم‌اند (مانند $w_{\max}$، $w_{\min}$، r_1، r_2، c_1 و c_2، نرخ جهش و...). پارامترهای مسئله مقدارهایی هستند که ویژگی‌های مسئله‌را نشان می‌دهند، مانند تعداد نوع سهم، کمینه بهره‌ی مورد انتظار و امثال آن... تعدادی از پارامترهای الگوریتم، عطف به تجربیات گذشته و آنچه در بخش ۳ آمده است، مقداردهی شده‌اند. مثلاً $w_{\max} = ۱/۰۵$، $w_{\min} = ۰/۴$ و $c_1 = c_2 = ۱/۴۹۶$. نرخ جهش از ۰/۵ تا مقدار نصف آن یعنی ۰/۲۵ به‌طور نزولی تغییر می‌کند. مناسب بودن مقدار ۰/۵ برای CBIPSO در مقالات متعددی گزارش شده است. به‌طور تجربی و براساس سعی و خطا، حد بالای سرعت برای قسمت صحیح ($V_{\max}I_d$) برابر با $\lfloor (ubd_d - lbd_d)/۴ \rfloor$ و برای قسمت دودویی ($V_{max}B_d$) برابر با $\lfloor (ubd_d - lbd_d)/۱۲ \rfloor$ قرار داده شده‌اند[۲۳].

برای تعیین مقدار بهینه‌ی پارامترهای مسئله، چند پارامتر دوبه‌دو تغییر داده شده و نتایج حاصل از اجرای الگوریتم با مقدارهای تنظیمی، ثبت و تحلیل شده‌اند. براساس این تحلیل، مقدارهای نسبتاً مناسب و کارآمد برای پارامترهای مورد آزمایش را استخراج، و حساسیت عملکرد الگوریتم CBIPSO را نیز نسبت به تغییر پارامترهای مختلف سنجیده‌ایم. پارامترهای مورد آزمایش عبارت‌اند از: اندازه‌ی مسئله (تعداد سهم‌های در دسترس)، اندازه‌ی سبد، کمینه نرخ بهره‌ی مورد انتظار، اندازه‌ی جمعیت و بیشترین تعداد ذره‌های ممکن. از این پنج پارامتر سه پارامتر نخست مربوط به مسئله و دو پارامتر آخر پارامترهای الگوریتم‌اند. معیارهای سنجش عملکرد CBIPSO میانگین ریسک‌ها و میانگین زمان اجرای الگوریتم هستند.

آزمایش‌ها نشان داده‌اند که بزرگ‌ترشدن مسئله (افزایش تعداد سهم‌های در دسترس) و نیز افزایش نرخ بهره‌ی مورد انتظار به‌طور کلی منجر به بزرگ‌تر شدن ریسک نهایی می‌شود. همین شرایط برای زمان اجرای الگوریتم برقرار است. به‌عبارت دیگر، برای رسیدن به جواب در مسئله‌های بزرگ‌تر باید زمان بیشتری صرف کرد که این امری است طبیعی. با افزایش اندازه‌ی جمعیت، میانگین ریسک‌ها علاوه بر آن که کاهش می‌یابد نسبت به تغییر در اندازه‌ی سبد نیز مقاوم‌تر می‌شود؛ یعنی نوسان آن نسبت به تغییر اندازه‌ی سبد کم می‌شود. این مطلب برای زمان اجرای الگوریتم

جدول ۴. خلاصه‌ی قیاس الگوریتم CBIPSO و الگوریتم ژنتیک.[۲۴]

	الگوریتم CBIPSO	الگوریتم ژنتیک
سرعت	بیشتر	کم‌تر
جواب	بهتر (ریسکِ کم‌تر)	بدتر (ریسکِ بیشتر)
تخصیص بودجه	بیش از ۹۷٪	۱۰۰٪
وضعیتِ گیر	ندارد	احتمال آن هست
جمعیت اولیه	لزوماً در ناحیه‌ی ممکن قرار ندارد	باید در ناحیه‌ی ممکن باشد
ثبات و تکرارپذیری جواب‌ها	بیشتر (انحراف معیار جواب‌ها کم‌تر)	کم‌تر (انحراف معیار جواب‌ها بیشتر)

صادق نیست، افزایش اندازه‌ی جمعیت ضمن افزودن بر زمان اجرا، آن را نسبت به تعداد سهم‌های درون سبد حساس‌تر می‌کند. چنین به نظر می‌رسد که باتوجه به ماهیت محاسبات موازی در CBIPSO، افزایش تعداد ذره‌ها باید بر سرعت آن بیفزاید. اما در یک برنامه‌ی رایانه‌یی، شبیه‌سازی الگوریتم با محاسبات متوالی انجام می‌شود و عملاً ذره‌ها یا عامل‌هایی[24] که مستقل از یکدیگر و هم‌زمان با هم به محاسبه بپردازند، به‌طور کامل وجود ندارند. از طرفی، کارهایی نظیر مقداردهی اولیه می‌تواند در افزایش زمان اجرای الگوریتم پرجمعیت‌تر مؤثر باشد. روشن شده است که عدد ۲۰ برای اندازه‌ی جمعیت مناسب است. برای تعداد موقعیت‌های ممکنِ بیش از ۳۰۰۰، به‌طور کلی تأثیر مهمی در جواب میانگین ظاهر نمی‌شود اما، بالطبع زمان اجرا زیاد می‌شود. تعداد موقعیت‌های ممکنِ ۳۰۰۰، مقداری نسبتاً مناسب است.

۳.۵. ثبات

نتایج نشان داده‌اند که متوسط انحراف معیار ریسک‌های CBIPSO(با معیار واریانس) به‌طور کلی از مرتبه‌ی 10^{-2} و سقف آن فقط ۰٫۱٪ است. این مطلب نشان می‌دهد که ریسک‌های به دست آمده از اجرای چندباره‌ی الگوریتم برای شرایط خاص، نوسان کمی داشته و ثبات و تکرارپذیری آن بالاست. بنابراین برای اندازه‌های متفاوتِ مسئله، سبد و نرخ بهره‌ی مورد انتظار، می‌توان از الگوریتم CBIPSO ثبات مناسبی توقع داشت.

۶. نتیجه‌گیری

الگوریتم PSO مبنای خوبی برای حل مسائل پیچیده و ترکیبی است، اما CBIPSO به‌عنوان شیوه‌ی گسترش‌یافته‌ی این الگوریتم پیشنهاد و معرفی شد. برای مسئله‌ی انتخاب سبد سهام طبق مدلی که ارائه شد، CBIPSO برتری خود را نسبت به الگوریتم ژنتیک نشان داد. در برخی موارد فاصله‌ی برتری CBIPSO از الگوریتم ژنتیک بسیار زیاد و چشم‌گیر، و در مواردی دیگر این فاصله نزدیک‌تر است. نتایج به‌کارگیری CBIPSO در حل مسئله ی انتخاب سبد سهام، که مسئله‌یی نسبتاً پیچیده به‌شمار می‌آید، نشان می‌دهد که از یک‌سو PSO هنوز رازهای نهفته‌یی دارد و می‌تواند در ابعاد گوناگونی گسترش یابد، و از سوی دیگر حل مسئله‌ی انتخاب سبد سهام در ابعاد واقعی و با لحاظ محدودیت‌های بازار، با استفاده از این روش بسیار مؤثر خواهد بود. نتایج حاصل از تحلیل حساسیت نیز نشان داد که اغلب پارامترهای CBIPSO را می‌توان به‌سادگی و براساس تجربیات گذشته در گونه‌های مختلف PSO که برای طیف وسیعی از مسائل مورد آزمایش قرار گرفته‌اند تنظیم کرد.

منابع

1. Markowitz, H.M. "Portfolio selection", *The Journal of Finance,* **7**, pp. 77-91, (March 1952).
2. Markowitz, H.M., *Portfolio Selection: Efficient Diversification of Investments,* New York: Wiley (1959).
3. Sharpe, W.F., "A simplified model for portfolio analysis", *Management Science,* **9**, pp. 277-293 (1963).
4. Mansini, R and Speranza, M. "Heuristic algorithms for the portfolio selection problem with minimum transaction lots", *European Journal of Operational Research,* **114**, pp. 219-233 (1999).
5. Mulvey, J.M. and Vladimirou, H. "Stochastic network programming for financial planning problems" *Management Science,* **38**, pp. 1642-1664 (1992).
6. Loraschi, A., et al., *Distributed Genetic Algorithms with an Application To Portfolio Selection,* Berlin: Springer Verlag, pp. 384-387 (1995), Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms (ICANNGA95).
7. Shoaf, J.S. and Foster, J.A., *A Genetic Algorithm Solution to the Efficient Set Problem: A Technique for Portfolio Selection Based on the Markowitz Model,* Dept. of Computer Science, Univ. of Idaho, Moscow: s.n., (1995). Tchnical Report TR-LAL-96-04.
8. Glover, F.; Mulvey, J.M. and Hoyland, K. *Solving Dynamic Stochastic Control Problems in Finance Using Tabu Search with Variable Scaling,* Meta-heuristics: theory & application, Springer, pp. 429-448 (1996).

پانوشت

1. combined binary and improved particle swarm optimization
2. mixed-integer quadratic programming (MIQP) problem
3. non-dominated sorting genetic algorithm II
4. turnover
5. trading constraint
6. multi-objective evolutionary algorithm (MOEA)
۷. این محدودیت ناشی از قوانین سرمایه‌گذاری کشور آلمان است: مقدار دارایی‌های مالی که از یک مصدر صادر می‌شوند، باید حداکثر تا ۵٪ ارزش خالص دارایی سهام شرکت سرمایه‌گذاری مشاع باشند؛ این مقدار می‌تواند تا ۱۰٪ باشد چنانچه، مجموع همه‌ی این دارایی‌ها کمتر از ۴۰٪ ارزش خالص دارایی باشد.
8. index tracking or sharpe's single factor model
9. asset allocation problem
10. reliability methods
۱۱. برای دریافت توضیح کاملی از مدل Markowitz و روش خط بحرانی به مرجع ۱ مراجعه کنید.
12. market or sector capitalization
13. classification
14. sector
15. class constraint
16. particle swarm optimization
17. position
18. personal best
19. global best
20. core leader
21. combined binary and improved particle swarm optimization
22. stability
۲۳. $[x]$ نشانه‌ی جزء صحیح یا کف عدد حقیقی x است.
24. Agent

9. Yoshimoto, A. "The mean-variance approach to portfolio optimization subject to transaction costs. 1", *Journal of Operations Research Society of Japan,* **15**, pp. 99-117 (1996).

10. Rolland, E., *A Tabu Search Method for Constrained Real-Number Search: Applications to Portfolio Selection,* Dept. of Accounting & Management Information Systems, Ohio State University, Columbus: s.n. (1996).

11. Chang, T.J., et al., "Heuristics for cardinality constrained portfolio optimisation", *Computers & Operations Research, Elsevier Science Ltd.,* **27**, pp. 1271-1302 (2000).

12. Busetti, F.R., *Metahuristic Approaches to Realistic Portfolio Optimization,* Ms Degree Dissertation, University of South Africa (2000).

13. Jobst, N.J., et al., "Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints", *Quantitative Finance,* **1**, pp. 489–501, Institute of Physics Publishing, (2001).

14. Schaerf, A. "Local search techniques for constrained portfolio selection problems", *Computational Economics,* **20**, pp. 177-190, Springer Netherlands, (December 2002).

15. Kellerer, H. and Maringer, D., *Optimization of Cardinality Constrained Portfolios with an Hybrid Local Search Algorithm,* Porto, Portugal: s.n., MIC'2001 - 4th Metaheuristics International Conference, pp. 585-589 (2001).

16. Lin, D. and Wang, S. "A genetic algorithm for portfolio selection problems", *Advanced Modeling and Optimization,* **4**, pp. 13-27 (2002).

17. Crama, Y. and Schyns, M., "Simulated annealing for complex portfolio selection problems", *Elsevier B.V., European Journal of Operational Research,* **150**, pp. 546–571 (2003).

18. Fieldsend, J.E.; Matatko, J. and Peng, M. "Cardinality constrained portfolio optimisation", *Intelligent Data Engineering and Automated Learning IDEAL, Berlin/Heidelberg Springer,* **3177**, pp. 788-793 (2004).

19. Streichert, F.; Ulmer, H. and Zell, A., *Evolutionary Algorithms and the Cardinality Constrained Portfolio Optimization Problem,* Operations research proceedings 2003: selected papers of the International Conference on Operations Research (OR 2003), Heidleberg, September 3-5, 2003, Heidleberg, pp. 253-260 (Springer 2004).

20. Streichert, F.; Ulmer, H. and Zell, A. "Evaluating a hybrid encoding and three crossover operators on the constrained portfolio selection problem", *IEEE, Proceedings of the Congress on Evolutionary Computation (CEC2004),* **1**, pp. 932-939 (2004).

21. Streichert, F.; Ulmer, H. and Zell, A. "Comparing discrete and continuous genotypes on the constrained portfolio selection problem", *Genetic and Evolutionary Computation – GECCO Berlin/Heidelberg,* **3103**, pp. 1239-1250 (Springer 2004).

22. Stein, M. ; Branke, J. and Schmeck, H., *Portfolio Selection: How to Integrate Complex Constraints,* (2005).

23. Oh, K.J., et al. "Portfolio algorithm based on portfolio beta using genetic algorithm", *Expert Systems with Applications, Elsevier Ltd.,* **30**, pp. 527-534 (2006).

24. Soleimani, H., *Portfolio Selection Using Genetic Algorithm,* Industrial Engineering Department, Amirkabir Univ. of Tech., MS Thesis, Supervisors: Golmakani H.R., Salimi Namin, M.H. (2007).

25. Soleimani, H.; Golmakani, H.R. and Salimi, M.H., "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm", *Expert Systems with Applications, Elsevier Ltd.,* **36**, pp. 5058-5063 (April 2009).

26. Zhang, W.G.; Chen, W. and Wang, Y.L. "The adaptive genetic algorithms for portfolio selection problem", *IJCSNS International Journal of Computer Science and Network Security,* **6**, pp. 196-200 (January 2006).

27. Lin, C.C. and Liu, Y.T. "Genetic algorithms for portfolio selection problems with minimum transaction lots", *Elsevier B.V., European Journal of Operational Research,* **185**, pp. 393-404 (2007).

28. Fernandez, A. and Gomez, S. "Portfolio selection using neural networks", *Elsevier Ltd., Computers & Operations Research,* **34**, pp. 1177-1191 (2007).

29. Xu, F.a.; Chen, W. and Yang, L., "Improved particle swarm optimization for realistic portfolio selection", *IEEE Computer Society,* Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 185-190 (2007).

30. Golmakani, H.R. and Jalilipour Alishah, E., "Portfolio selection using an artificial Immune system", *Las Vegas: IEEE, IRI 2008 (Information Reuse Integration),* pp. 28-33 (2008).

31. Hanafizadeh, P. and Ponnambalam, K. "Asset allocation using reliability method", *Elsevier Ltd., Mathematical and Computer Modelling,* **50**, pp. 21-31 (2009).

32. Elbeltagi, E.; Hegazy, T. and Grierson, D. "Comparison among five evolutionary-based optimization algorithms", *Advanced Engineering Informatics,* **19**, pp. 43-53 (2005).

33. Hassan, R.; Cohanim, B. and de Weck, O., *A Comparison of Particle Swarm Optimization and the Genetic Algorithm,* In Proceedings of the 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference (2005).

34. Mansini, R. and Sperznza, M.G. "An exact approach for portfolio selection with transaction costs and round", *IIE Transactions,* **37**, pp. 919-929, (2005), ISSN 1545-8830 online.

35. Kennedy, J. and Eberhart, R. "Particle swarm optimization", *IEEE, Proceeding of International Conference on Neural Networks,* pp. 1942-1948, (1995), 0-7803-2768-3.

36. Kennedy, J. and Eberhart, R. "A new optimizer using particle swarm theory", *IEEE, Sixth International Symposium on Micro Machine and Human Science,* pp. 39-43, (1995), 0-7803-2676-8.

37. Kennedy, J. "The particle swarm: Social adaptation of knowledge", *IEEE,* pp. 303-308 (1997).

38. Shi, Y. and Eberhart, R. "A modified particle swarm optimizer", *IEEE,* pp. 69-73 (1998).

39. Li, X.E. and Andries, P., *Particle Swarm Optimization-an Introduction and its Recent Developments (A tutorial),* London: ACM, GECCO'07, pp. 3391-3414 (2007).

40. Kennedy, J. and Eberhart, R. "A discrete binary version of particle swarm algorithm", *IEEE*, pp. 4104-4108 (1997), 0-7803-4053-1.

41. Coath, G. and Halgamuge, S.K. "A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems", *IEEE*, pp. 2419-2425 (2003), 0-7803-7804-0.

42. Zheng, J.; Wu, Q. and Song, W. "An improved particle swarm algorithm for solving nonlinear constrained optimization problems", *IEEE-Computer Society,* Third International Conference on Natural Computation (ICNC 2007), 0-7695-2875-9.

43. Pulido, G.T. and Coello Coello, C.A. "A constraint-handling mechanism for particle swarm optimization", *IEEE*, pp. 1396-1403, (2004), 0-7803-8515-2.

44. Eberhart, R.C. and Shi, Y. "Comparing inertia weights and constriction factors in particle swarm optimization", *IEEE*, pp. 84-88, (2000), 0-7803-6375-2.

45. Coello Coello, C.A.; Pulido, G.T. and Lechuga, M.S. "Handling multiple objectives with particle swarm optimization", *IEEE, Transactions on Evolutionary Computation,* **8**, pp. 256-279, (June 2004), 1089-778X.

46. Fazel, M., *Portfolio Selection Using Particle Swarm Optimization,* Industrial Engineering Department, Tafresh Univ., MS Thesis, Supervisor: Golmakani, Hamid Reza (2009).