مهندسی صنایع و مدیریت شریف (تابستان ۱۳۹۵)
دوره‌ی ۱ - ۳۲، شماره‌ی ۱/۱، ص. ۷۹-۸۷

Original Article

مدل‌سازی و زمان‌بندی مسئله‌ی جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی

صفیه گلچین (دانشجوی کارشناسی ارشد)
مانی شریفی (استادیار)
دانشکده‌ی مهندسی صنایع و مکانیک، دانشگاه آزاد اسلامی واحد قزوین
بهمن نادری * (استادیار)
گروه مهندسی صنایع، دانشگاه خوارزمی

در این پژوهش مسئله‌ی زمان‌بندی جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی، با هدف کمینه کردن زمان تکمیل کل کارها مورد مطالعه قرار گرفته است. ماشین‌های پردازش دسته‌یی از قابلیت پردازش همزمان چند کار در یک دسته برخوردارند. ظرفیت ماشین‌ها و اندازه‌ی کارها در هر مرحله مشخص است. دسته‌ها پس از تشکیل تا آخرین مرحله ثابت می‌مانند. مجموع اندازه کارهای هر دسته نباید از کوچک‌ترین ظرفیت ماشین‌ها بیشتر شود. زمان پردازش دسته‌ها برابر طولانی‌ترین زمان پردازش کارها در دسته است. ابتدا مدل برنامه‌ریزی خطی عدد صحیح مختلط برای مسئله‌ی مورد نظر پیشنهاد داده می‌شود. به‌دلیل پیچیدگی بالای مسئله‌ی مورد بررسی، الگوریتم فرا‌ابتکاری رقابت استعماری برای حل مسئله توسعه داده شده است. در نهایت عملکرد الگوریتم پیشنهادی در برابر الگوریتم‌های شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات موجود در ادبیات، مورد بررسی قرار گرفته است. نتایج نشان می‌دهد که الگوریتم رقابت استعماری نسبت به دو الگوریتم دیگر برای مسئله‌ی مورد نظر عملکرد بهتری دارد.

واژگان کلیدی: زمان‌بندی جریان کارگاهی ترکیبی، ماشین‌های پردازش دسته‌یی، برنامه‌ریزی خطی عدد صحیح مختلط، الگوریتم رقابت استعماری.

golchin.safieh@yahoo.com
m.sharifi@qiau.ac.ir
bahman.naderi@aut.ac.ir

۱. مقدمه

در دهه‌ی اخیر به دسته‌بندی کارها در مسائل زمان‌بندی ــ نظیر تحویل کالا به‌صورت دسته‌یی یا ورود دسته‌یی کارها به سیستم ــ توجه خاصی شده است. یکی از بهترین پژوهش‌های انجام شده در این زمینه پردازش دسته‌یی کارهاست، زیرا در این حالت پردازش کارها ارزان‌تر و سریع‌تر از پردازش آنها به‌صورت تکی است. برای انجام عملیات به‌صورت دسته‌یی از ماشین‌های پردازش دسته‌یی[1] در محیط‌های تولیدی استفاده شده است. ازجمله مزایای استفاده از این ماشین‌ها کاهش زمان تولید، کاهش زمان آماده‌سازی، کاهش هزینه‌ی حمل و نقل، و کاهش نیروی انسانی است. لذا زمان‌بندی ماشین‌های پردازش دسته‌یی می‌تواند سهم عمده‌یی در بهبود و پیشرفت عملکرد شرکت‌ها داشته باشد.

لازم به ذکر است که امکان پردازش دسته‌یی تنها در بعضی از صنایع وجود دارد. ماشین‌های پردازش دسته‌یی از قابلیت پردازش همزمان چندکار در یک دسته برخوردارند و در صنایع مختلف نظیر صنایع الکترونیک برای تولید برد مدار چاپی[2]، صنایع فلزکاری[3] و همچنین کوره‌های پخت کاربرد دارند. هر ماشین می‌تواند دسته‌یی از کارها را به‌طوری که ظرفیت دسته از ظرفیت ماشین بیشتر نشود، پردازش کند. با توجه به کاربرد ماشین‌ها زمان پردازش دسته‌ها ثابت یا متغیر است. ثابت بودن زمان پردازش دسته‌ها به‌منزله‌ی محاسبه‌ی زمان مستقل از کارهای درون دسته است. متغیر بودن زمان پردازش دسته‌ها نیز به‌منزله‌ی وابستگی زمان به کارهای درون دسته است که براساس مجموع زمان پردازش کارها یا بیشینه زمان پردازش کارهای درون دسته محاسبه می‌شود. ظرفیت دسته‌های تشکیل شده می‌تواند نامحدود، یا محدود به تعداد کارها یا اندازه و حجم آنها باشد.

زمان‌بندی، تعیین توالی کارها برای تخصیص به منابع تولید است که به‌منظور رسیدن به اهدافی مانند بهره‌برداری بهتر از منابع دردسترس، پاسخگویی سریع به تقاضای مشتریان، و کمینه کردن زمان تکمیل کل کارها انجام می‌شود. مسائل زمان‌بندی را می‌توان با توجه به تعداد عملیات لازم برای پردازش یک کار و نیز تعداد ماشین‌های موجود برای انجام هر عملیات، از هم تفکیک کرد. محیط جریان کارگاهی شامل چند مرحله‌ی متوالی است که در هر مرحله عملیات خاصی روی کارها انجام می‌شود. کارها برای پردازش از تمامی مراحل به‌ترتیب باید عبور کنند. با توجه به شرایط رقابتی صنایع، بسیاری از سیستم‌های تولیدی کلاسیک

* نویسنده مسئول
تاریخ: دریافت ۱۳۹۲/۱۲/۱۰، اصلاحیه ۱۳۹۳/۴/۲، پذیرش ۱۳۹۳/۷/۱۵.

به سمت محیط‌های جدیدتر مانند جریان کارگاهی ترکیبی[۴] تغییر جهت می‌دهند. این سیستم تولیدی ترکیبی از جریان کارگاهی و ماشین‌های موازی است. سیستم جریان کارگاهی، به‌منظور افزایش ظرفیت خود در مراحل گلوگاه از ماشین‌های موازی استفاده می‌کند. ماشین‌های موازی می‌توانند یکسان، یکنواخت یا نامرتبط باشند. در سیستم جریان کارگاهی ترکیبی حداقل در یک مرحله بیش از یک ماشین وجود دارد.

از آنجا که در تمامی محیط‌های زمان‌بندی به‌جای ماشین‌های پردازش تکی، از ماشین‌های پردازش دسته‌یی می‌توان استفاده کرد، پژوهش‌های انجام شده برمبنای محیط‌های زمان‌بندی طبقه‌بندی می‌شود. در طرح پیچیدگی مسئله‌ی زمان‌بندی با یک ماشین پردازش دسته‌یی، برای حالتی که اندازه دسته‌ها نامحدود یا محدود به تعداد کار است،[۱] بعد از ایجاد یک ترتیب بهینه برای دسته‌ها از الگوریتم‌های برنامه‌ریزی پویا برای حل مسئله استفاده شد که با نماد $1/batch/L_{\max}$ نشان داده می‌شود. در بررسی مسئله‌ی تک‌ماشینی پردازش دسته‌یی، با یکسان نبودن اندازه کارها[۲] زمان پردازش، اندازه و ظرفیت هریک از ماشین‌ها مشخص بوده است. زمان پردازش دسته‌ها برابر طولانی‌ترین زمان پردازش کارهای درون دسته است. در این پژوهش برای حل مسئله از الگوریتم ژنتیک با هدف کمینه‌کردن زمان تکمیل کل کارها استفاده شده است. برای ارزیابی الگوریتم پیشنهادی، مثال‌هایی به‌صورت تصادفی تولید شده و نتایج به‌دست آمده از الگوریتم ژنتیک، با الگوریتم شبیه‌سازی تبرید و نرم‌افزار تجاری حل مسئله مقایسه شده است. نتایج مقایسه‌ها، نشان‌گر آن است که الگوریتم ژنتیک تابع هدف بهتری در زمان اجرای کم‌تری به‌دست می‌آورد که با نماد $1/batch/C_{\max}$ نشان داده می‌شود.

پژوهش‌های انجام شده در زمینه‌ی پردازش دسته‌یی کارها در محیط جریان کارگاهی را می‌توان به دو بخش تقسیم کرد: ۱. پردازش دسته‌یی کارها در محیط جریان کارگاهی دومرحله‌یی، ۲. پردازش دسته‌یی کارها در محیط جریان کارگاهی چندمرحله‌یی.

در حوزه‌ی زمان‌بندی دسته‌یی در محیط جریان کارگاهی دومرحله‌یی، محققین مسئله را در حالتی بررسی کرده‌اند که در مرحله‌ی اول کارها به‌صورت تکی و در مرحله‌ی دوم به‌صورت دسته‌یی پردازش شده است.[۳] آنان زمان پردازش دسته را بدون توجه به کارهای موجود در دسته ثابت در نظر گرفته و یک الگوریتم ابتکاری برای حل مسئله با هدف کمینه‌کردن زمان تکمیل کل کارها ارائه داده‌اند که با نماد $F2/batch(II)/C_{\max}$ نشان داده می‌شود. در پژوهشی دیگر،[۴] محققین دو مدل برنامه‌ریزی خطی عدد صحیح مختلط برای مسئله‌ی زمان‌بندی جریان کارگاهی دومرحله‌یی با ماشین‌های پردازش دسته‌یی ارائه شده است. آنها در پژوهش خود ابتدا با فرض صفربودن ظرفیت انبار میانی، مسئله را مدل‌سازی کرده و مدل دیگری ارائه داده‌اند که ظرفیت انبار میانی آن نامحدود باشد. همچنین آنها زمان پردازش دسته‌ها را برابر طولانی‌ترین زمان پردازش کارهای دسته فرض کرده‌اند. در پژوهش یادشده ابتدا کارها را دسته‌بندی و سپس توالی دسته‌ها را تعیین می‌کنند. برای حل مسئله با اندازه کوچک نرم‌افزار سیپلکس (CPLEX) پیشنهاد شده که با نماد $F2/batch/C_{\max}$ نشان داده می‌شود.

دربررسی مسئله‌ی زمان‌بندی جریان کارگاهی دومرحله‌یی با ماشین‌های پردازش دسته‌یی،[۵] ابتدا مدل برنامه‌ریزی خطی عدد صحیح مختلط که در سال ۲۰۰۴ با هدف کمینه‌کردن زمان تکمیل کل کارها ارائه شد، بهبود یافت. در پژوهش یادشده دسته‌بندی کارها و تعیین توالی دسته‌ها، دو تصمیم وابسته به هم بیان شده است. در نهایت، یک کران پایین و یک الگوریتم ابتکاری براساس برنامه‌ریزی خطی عدد صحیح مختلط برای حل مسئله ارائه شده که با نماد $F2/batch/C_{\max}$ نشان داده می‌شود. در بررسی مسئله‌ی زمان‌بندی جریان کارگاهی دومرحله‌یی با ماشین‌های پردازش دسته‌یی، به‌طوری که بین دو مرحله دسته‌ها منتظر نمانند، زمان پردازش کارها در هر مرحله، زمان در دسترس بودن کارها و نیز اندازه کارها مشخص است.[۶،۷] زمان پردازش دسته برای ماشین اول برابر طولانی‌ترین زمان پردازش کارها در دسته، و برای ماشین دوم برابر جمع زمان‌های پردازش همه کارهای دسته است. یک مدل برنامه‌ریزی خطی عدد صحیح مختلط برای مسئله‌ی مورد نظر ارائه شده است. الگوریتم فراابتکاری بهینه‌سازی ازدحام ذرات برای حل مسئله پیشنهاد شده که با نماد $F2/batch,nwt,r_j/C_{\max}$ نشان داده می‌شود. محققین براساس الگوریتم جست‌وجوی ممنوعه، یک الگوریتم ابتکاری برای حل مسئله‌ی زمان‌بندی جریان کارگاهی دومرحله‌یی با ماشین‌های پردازش دسته‌یی بررسی کرده‌اند[۸] که هدف آن کمینه‌کردن زمان تکمیل کل کارها بوده است. الگوریتم پیشنهادی را با یک الگوریتم ابتکاری براساس برنامه‌ریزی خطی عدد صحیح مختلط مقایسه می‌کنند که با نماد $F2/batch/C_{\max}$ نشان داده می‌شود.

در مطالعه‌ی انجام شده درخصوص مسئله‌ی زمان‌بندی جریان کارگاهی چندمرحله‌یی با ماشین‌های پردازش دسته‌یی،[۹] زمان پردازش دسته‌ها برابر طولانی‌ترین زمان پردازش کارهای دسته است. ظرفیت ماشین در همه‌ی مراحل ثابت فرض شده و برای حل مسئله با هدف کمینه‌کردن کل دیرکرد، بیشترین دیرکرد و تعداد کارهای با تأخیر الگوریتم جست‌وجوی همسایگی متغیر پیشنهاد شده که با نماد $F_m/batch/C_{\max}$ نشان داده می‌شود. در مطالعه‌یی دیگر، محققین در بررسی مسئله‌ی زمان‌بندی جریان کارگاهی چندمرحله‌یی با ماشین‌های پردازش دسته‌یی،[۱۰] مدل ارائه‌شده در سال ۲۰۰۴ را بهبود دادند. بدین‌ترتیب آنان پنج کران پایین برای حل مسئله ارائه داده‌اند که با نماد $F_m/batch/C_{\max}$ نشان داده می‌شود. در پژوهشی دیگر درخصوص مسئله‌ی زمان‌بندی جریان کارگاهی چندمرحله‌یی با ماشین‌های پردازش دسته‌یی،[۱۱] محققین زمان پردازش دسته‌ها را برابر طولانی‌ترین زمان پردازش کارهای در دسته فرض کرده و کارها را طوری دسته‌بندی می‌کنند که ظرفیت دسته‌ها از ظرفیت ماشین‌ها بیشتر نشود. آنان الگوریتم فراابتکاری بهینه‌سازی ازدحام ذرات را برای حل مسئله با هدف کمینه‌کردن زمان تکمیل کل کارها ارائه داده‌اند که با نماد $F_m/batch,prmu/C_{\max}$ نشان داده می‌شود.

در بررسی مسئله‌ی زمان‌بندی دسته‌های موازی در محیط جریان کارگاهی ترکیبی با هدف کمینه‌کردن زمان تکمیل کل کارها،[۱۲] با این فرض که در بعضی از مراحل، ماشین‌ها توانایی پردازش هم‌زمان دو یا تعداد بیشتری ازکارها را دارند (اگر ظرفیت ماشین برابر ۱ باشد، ماشین فقط یک کار را پردازش می‌کند)، سه الگوریتم ابتکاری برای حل مسئله ارائه شده و یک کران پایین نیز برای ارزیابی الگوریتم‌های پیشنهادی در نظرگرفته شده است. در نهایت الگوریتم ژنتیک سه‌بعدی برای بالا بردن کیفیت جواب‌ها پیشنهاد شده که با نماد $FF_c/p-batch/C_{\max}$ نشان داده می‌شود.

با توجه به مطالعات انجام شده، به مسائل زمان‌بندی دسته‌یی کارها در محیط جریان کارگاهی ترکیبی کم‌تر توجه شده است. از آنجا که محیط‌های صنعتی به سمت جریان کارگاهی ترکیبی در حال تغییرند، پردازش دسته‌یی کارها برای استفاده‌ی بهینه از ظرفیت ماشین‌ها مورد توجه است. هدف این پژوهش، ارائه‌ی مدلی برای پردازش کارها به‌صورت دسته‌یی در محیط جریان کارگاهی ترکیبی با هدف کمینه‌کردن زمان تکمیل کل کارهاست که با نماد $FF_c/batch,P_m/C_{\max}$ نشان داده می‌شود.

۲. تعریف مسئله

ویژگی مسئله‌ی زمان‌بندی مورد بررسی در این پژوهش عبارت است از: n کار که ابتدای افق زمان‌بندی در دسترس‌اند و روی c مرحله باید پردازش شوند. در هر مرحله‌ی l، تعداد m_l ماشین به‌صورت موازی وجود دارد. تمامی کارها از تمامی مراحل باید عبور کنند و در هر مرحله فقط روی یک ماشین پردازش شود. مسیر پردازش برای تمامی کارها یکسان است. دسته‌ها فقط یک بار پردازش می‌شوند؛ به‌عبارت دیگر هنگامی که پردازش یک دسته شروع شد امکان قطع عملیات وجود ندارد. زمان پردازش هر دسته برابر بزرگ‌ترین زمان پردازش کارهای دسته است. ماشین‌های موازی در هر مرحله یکسان و ظرفیت آنها مشخص است. مجموع اندازه کارهایی که در یک دسته قرار می‌گیرد نباید از ظرفیت ماشین‌ها بیشتر شود. هنگامی که دسته‌ها تشکیل شدند تا آخرین مرحله ثابت می‌مانند و ظرفیت دسته‌ها با کم‌ترین ظرفیت ماشین‌ها در نظر گرفته می‌شود.

۳. مدل ریاضی پیشنهادی

در این بخش مدل برنامه‌ریزی خطی عدد صحیح مختلط برای مسئله‌ی مورد بررسی ارائه می‌شود. مدل پیشنهادی، توسعه‌یافته‌ی مدل ارائه شده برای محیط جریان کارگاهی است.[۱۱] پارامترها و نمادهای استفاده شده در مدل چنین توضیح داده شده است:

n: تعداد کارها؛

c: تعداد مراحل مختلف کاری؛

B: تعداد دسته‌ها؛

m_l: تعداد ماشین‌های موازی در مرحله‌ی l؛

s_j: اندازه کار j؛

p_{lj}: زمان پردازش کار j در مرحله‌ی l؛

j: اندیس کار $j = ۱, ۲, ..., n$؛

l: اندیس مرحله $l = ۱, ۲, ..., c$؛

b, k: اندیس‌های دسته $b, k = ۱, ۲, ..., B$؛

S_c ظرفیت ماشین مرحله‌ی c؛

M: عدد بزرگ مثبت.

لازم به ذکر است که حد بالای تعداد دسته‌ها را معمولاً به اندازه تعداد کارها در نظر می‌گیرند. همچنین متغیرهای تصمیم مدل عبارت‌اند از:

$X_{b,l,k}$: ۱، اگر دسته‌ی b بعد از دسته‌ی k در مرحله‌ی l پردازش شود؛ ۰، در غیر این‌صورت.

$Y_{b,l,i}$: ۱، اگر دسته‌ی b در مرحله‌ی l روی ماشین i پردازش شود؛ ۰، در غیر این‌صورت.

$U_{b,l,k}$: ۱، اگر دسته‌ی b و دسته‌ی k روی ماشین یکسان در مرحله‌ی l پردازش شود؛ ۰، در غیر این‌صورت.

$Z_{j,b}$: ۱، اگر کار j در دسته‌ی b قرار گیرد؛ ۰، در غیر این‌صورت.

$C_{l,b}$: متغیری پیوسته است که زمان تکمیل دسته b در مرحله l را نشان می‌دهد.

$$\text{Min } C_{\max} \qquad (۱)$$

Subject to :

$$\sum_{i=۱}^{m_l} Y_{b,l,i} = ۱ \qquad \forall b, l \qquad (۲)$$

$$\sum_{b=۱}^{B} Z_{j,b} = ۱ \qquad \forall j \qquad (۳)$$

$$C_{۱,b} \geq Z_{j,b} p_{۱,j} \qquad \forall b, j \qquad (۴)$$

$$\sum_{j=۱}^{n} S_j Z_{j,b} \leq \min(S_۱, S_۲, ..., S_c) \qquad \forall b \qquad (۵)$$

$$C_{l,b} \geq C_{l-۱,b} + Z_{j,b} p_{l,j} \qquad \forall b, j, l > ۱ \qquad (۶)$$

$$C_{l,b} - C_{l,k} + M.(۱ - X_{b,l,k}) \geq Z_{j,b} p_{l,j} \qquad \forall b < B, \quad k > b, j, l \qquad (۷)$$

$$C_{l,k} - C_{l,b} + M.(۱ - U_{b,l,k} + X_{b,l,k}) \geq Z_{j,k} p_{l,j} \qquad \forall b < B, \quad k > b, j, l \qquad (۸)$$

$$U_{b,l,k} \geq Y_{b,l,i} + Y_{k,l,i} - ۱ \qquad \forall b < B, \quad k > b, l, i \qquad (۹)$$

$$C_{\max} \geq C_{c,b} \qquad \forall j$$

$$C_{l,b} \geq ۰ \qquad \forall l, b$$

$$Z_{j,b} \in \{۰, ۱\} \qquad \forall j, b$$

$$X_{b,l,k} \in \{۰, ۱\} \qquad \forall b, l, k > b$$

$$Y_{b,l,i} \in \{۰, ۱\} \qquad \forall b, l, i \qquad (۱۰)$$

تابع هدف، کمینه‌کردن زمان تکمیل کل کارهاست که در معادله‌ی ۱ نشان داده شده است. دسته محدودیت ۲ نشان می‌دهد که هر دسته در هر مرحله فقط روی یک ماشین پردازش شود. دسته محدودیت ۳ نشان‌دهنده‌ی آن است که هر کار فقط به یک دسته تخصیص داده شود. دسته محدودیت ۴ زمان تکمیل دسته‌ها در مرحله‌ی اول را مشخص می‌کند. دسته محدودیت ۵ این اطمینان را می‌دهد که ظرفیت دسته‌ها از ظرفیت ماشین‌ها بیشتر نشود. دسته محدودیت ۶ زمان تکمیل دسته‌ها را از مرحله‌ی دو تا آخرین مرحله مشخص می‌کند که با زمان تکمیل هر دسته در مرحله‌ی قبل به اضافه‌ی زمان پردازش دسته در مرحله‌ی مورد نظر برابر است. دسته محدودیت‌های ۷ و ۸ مربوط به هر جفت از دسته‌ها هستند که تضمین می‌کنند هر ماشین در هر مرحله و در هر زمان حداکثر در حال پردازش یک دسته است. برای هر جفت از دسته‌ها فقط یکی از این دو دسته محدودیت‌ها فعال می‌شود. دسته محدودیت ۹ تعیین‌کننده‌ی کارهایی است که در مرحله‌ی l روی ماشین یکسان پردازش شده است. دسته محدودیت ۱۰ مقدار تابع هدف را محاسبه می‌کند.

مسئله‌ی جریان کارگاهی با پردازش دسته‌یی یک مسئله‌ی NP-hard است.[۱۱] مسئله‌ی جریان کارگاهی یک حالت خاص از مسئله‌ی جریان کارگاهی ترکیبی است، و درنتیجه مسئله‌ی ذکر شده نیز NP-hard خواهد بود. پیدا کردن جواب بهینه در زمان معقول برای مسائل در اندازه‌ی بزرگ امکان‌پذیر نیست. بنابراین استفاده از الگوریتم فراابتکاری برای حل مسئله در اندازه‌های مختلف ضرورت پیدا می‌کند.

۴. الگوریتم رقابت استعماری

اکثر روش‌های بهینه‌سازی شناخته شده، شبیه‌سازی رایانه‌یی فرایندهای طبیعی هستند. در ارائه‌ی الگوریتم‌های بهینه‌سازی علاوه بر توجه به تکامل زیستی انسان و سایر موجودات به تکامل اجتماعی و تاریخی او به‌عنوان پیچیده‌ترین و موفق‌ترین حالت

تکامل، توجه چندانی نشده است. الگوریتم رقابت استعماری[5] در سال ۲۰۰۷ برای اولین بار توسط آتش‌پز ـ گرگری و لوکاس ارائه شده است.[۱۳]

۱.۴. مرحله‌ی آغازین

اولین مرحله‌ی الگوریتم شامل نحوه‌ی نمایش جواب، تولید جمعیت اولیه و شکل‌دهی امپراطوری‌های اولیه است. نحوه‌ی نمایش جواب به‌صورت یک بردار n بعدی است (n نشان‌دهنده‌ی تعداد کارهاست). تولید جمعیت اولیه و شکل‌دهی امپراطوری‌های اولیه در ادامه توضیح داده می‌شوند.

۱.۱.۴. تولید جمعیت اولیه

برای تولید جمعیت اولیه الگوریتم رقابت استعماری از الگوریتم ابتکاری First Fit استفاده شده است. مسئله‌یی با تعداد n کار را در نظر بگیرید. ابتدا ترتیبی تصادفی متشکل از اعداد صحیح ۱ تا n ایجاد می‌شود که هر عدد متناظر با یک کار است. از الگوریتم ابتکاری First Fit برای دسته‌بندی کارها استفاده می‌شود. در این الگوریتم کارها به‌ترتیب از چپ به راست انتخاب شده و ظرفیت دسته‌ها به‌ترتیب شماره از کوچک به بزرگ بررسی می‌شود. اولین دسته‌یی که ظرفیت آن بیشتر از ظرفیت کار مورد نظر باشد، انتخاب شده و کار را در آن دسته قرار می‌دهیم. به همین ترتیب ادامه می‌دهیم تا تمامی کارها دسته‌بندی شود.

۲.۱.۴. شکل‌دهی امپراطوری‌های اولیه

برای شروع الگوریتم، کشورهای اولیه براساس الگوریتم ابتکاری First Fit تولید می‌شود. تعداد این کشورها برابر اندازه جمعیت است. اندازه جمعیت از پارامترهای ورودی الگوریتم است. سپس تابع هزینه‌ی هریک از کشورها محاسبه می‌شود. دسته‌ها به‌ترتیب شماره و براساس دسترس‌پذیری ماشین‌ها در مرحله‌ی اول پردازش می‌شوند و از مرحله‌ی دو تا آخرین مرحله دسته‌ها بر اساس زمان تکمیل مرحله‌ی قبل پردازش می‌شوند. به تعداد امپراطوری، از بهترین اعضای این جمعیت که دارای کم‌ترین مقدار تابع هزینه هستند به‌عنوان استعمارگر انتخاب می‌شوند. تعداد امپراطوری از پارامترهای ورودی الگوریتم است. جمعیت باقی‌مانده به‌عنوان کشورهای مستعمره هستند که هرکدام به یک امپراطوری تعلق دارند. برای تقسیم مستعمرات اولیه بین استعمارگرها از انتخاب چرخه‌ی رولت استفاده شده است. برای انجام این کار با داشتن هزینه‌ی تمامی استعمارگرها، هزینه‌ی هنجارشده‌ی آنها براساس معادله‌ی ۱۱ محاسبه می‌شود.

$$C_n = \exp(\frac{-c_n}{\max c_i}) \qquad i = ۱, ۲, ..., nEmp \qquad (۱۱)$$

c_n هزینه‌ی nامین استعمارگر است، C_n هزینه‌ی هنجارشده‌ی nامین استعمارگر است و $\max c_i$ بیشینه هزینه در بین تمام استعمارگرهاست. $nEmp$ تعداد امپراطوری‌هاست. با داشتن هزینه‌ی هنجارشده، قدرت نسبی به‌هنجار هر استعمارگر براساس معادله‌ی ۱۲ محاسبه می‌شود و مبنایی برای تقسیم کشورهای مستعمره بین استعمارگرهاست.

$$p_n = \frac{C_n}{\sum_{i=۱}^{nEmp} C_i}, \qquad \sum_{i=۱}^{nEmp} p_i = ۱ \qquad (۱۲)$$

روش چرخه‌ی رولت[6] یکی از معروف‌ترین روش‌های انتخابی است. در این روش ابتدا تمام مقادیر احتمال انتخاب در کنار یکدیگر چیده شده و سپس یک عدد تصادفی در بازه‌ی صفر تا ۱ تولید می‌شود. انتخاب این بازه به این دلیل است که مجموع مقادیر احتمال انتخاب، همیشه برابر با ۱ خواهد بود. از مقایسه‌ی عدد تصادفی با بازه چرخه رولت، شماره‌ی استعمارگر متناظر با عدد تصادفی مشخص می‌شود. از آنجا که مقدار احتمال هر استعمارگر، بخشی از فضای چرخ رولت را به خود اختصاص داده است احتمال انتخاب استعمارگرهای شایسته‌تر (دارای تابع هزینه‌ی کم‌تر) بیشتر خواهد بود. با این روش تمام کشورهای مستعمره به استعمارگرها تخصیص داده می‌شود و با داشتن حالت اولیه‌ی تمام امپراطوری‌ها، الگوریتم رقابت استعماری شروع می‌شود. روند تکامل در یک حلقه قرار دارد که تا برآورده شدن شرط توقف ادامه می‌یابد.

۲.۴. عملگر جذب

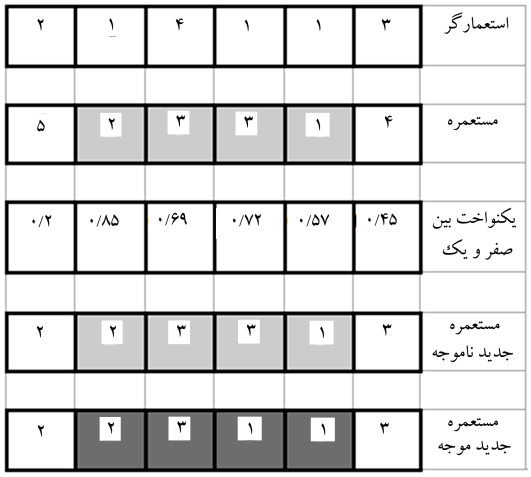
در هر امپراطوری، کشور استعمارگر به‌منظور افزایش نفوذ خود سعی می‌کند تعداد مستعمره‌هایش را افزایش دهد. لذا در هر امپراطوری کشورهای مستعمره شروع به حرکت به سمت استعمارگر مربوطه می‌کنند. این بخش از فرایند استعمار، در الگوریتم بهینه‌سازی به‌صورت حرکت مستعمرات به سمت کشور استعمارگر مدل شده است. در حقیقت حکومت مرکزی با اعمال سیاست جذب سعی دارد کشور مستعمره را در راستای ابعاد مختلف اجتماعی ـ سیاسی به خود نزدیک کند. در الگوریتم ارائه شده سیاست جذب چنین تعریف شده است:

با توجه به ابعاد کشور مستعمره که برابر تعداد کارهاست، ابتدا رشته‌یی از اعداد به‌صورت یکنواخت بین صفر و ۱ تولید می‌کنیم. سپس سلول‌هایی را که عدد متناظر با آن کوچک‌تر از ۰/۵ است مشخص می‌کنیم. این سلول‌ها متناظر با شماره کارها در کشور مستعمره است. کارهای مشخص شده را بر اساس کشور استعمارگر دسته‌بندی می‌کنیم. سایر کارهای کشور مستعمره را چنان‌که قبلاً دسته‌بندی شده بود، دسته‌بندی می‌کنیم. با توجه به مثالی که در بخش تولید جمعیت اولیه در نظرگرفته شد سیاست جذب در شکل ۱ نشان داده شده است.

همان‌طور که در شکل ۱ مشخص است در مستعمره‌ی جدید کارها در چهار دسته تقسیم‌بندی شده است که ظرفیت دسته‌ها به‌ترتیب پنج، هفت، هشت و چهار است و هیچ‌یک از ظرفیت مجاز هشت بیشتر نشده است. اگر جواب به‌دست آمده موجه بود به مرحله‌ی بعد می‌رویم و در غیراین‌صورت، بعد از این که کارهای مشخص شده‌ی کشور مستعمره را مطابق کشور استعمارگر دسته‌بندی کردیم سایر کارهای کشور مستعمره را نیز با استفاده از الگوریتم ابتکاری Best Fit دسته‌بندی می‌کنیم. منظور از موجه بودن این است که ظرفیت دسته‌های تشکیل شده از حد مجاز عبور نکرده باشد. در الگوریتم ابتکاری Best Fit سعی می‌شود از تمامی ظرفیت دسته‌های تشکیل شده، استفاده شود. بنابراین کارهای دسته‌بندی نشده به‌ترتیب از

استعمارگر	۳	۱	۱	۴	۱	۲
مستعمره	۴	۱	۳	۳	۲	۵
یکنواخت بین صفر و یک	۰/۵۵	۰/۳۷	۰/۹۸	۰/۶۵	۰/۸۵	۰/۰۲
مستعمره جدید	۴	۱	۳	۳	۲	۲

شکل ۱. اجرای عملیات انتقال در صورت موجه بودن جواب.



شکل ۲. اجرای عملیات انتقال در صورت ناموجه بودن جواب.

چپ انتخاب، و ظرفیت دسته‌ها به‌طور صعودی مرتب می‌شود. اولین دسته‌یی که ظرفیت باقی‌مانده‌ی آن مساوی یا بزرگ‌تر از ظرفیت کار مورد نظر است انتخاب می‌شود و کار را در آن دسته قرار می‌دهیم.

چنان که در شکل ۲ مشاهده می‌شود، بعد از عملیات جذب، دسته‌هایی که در مستعمره‌ی جدید تشکیل شده دارای ظرفیت‌هایی به‌ترتیب برابر ۱، ۸ و ۱۱ است که ظرفیت دسته‌ی سه از ظرفیت مجاز بیشتر شده است. لذا ابتدا کارهای یک و شش را دسته‌بندی کرده و سپس به‌ترتیب کارهای دو تا چهار را که با رنگ تیره‌تر مشخص شده توسط الگوریتم Best Fit دسته‌بندی می‌کنیم. بنابراین در مستعمره‌ی جدید کارها در چهار دسته، تقسیم‌بندی شده که ظرفیت دسته‌ها به‌ترتیب برابر پنج، هشت، هفت و چهار است که هیچ یک از هشت بیشتر نشده است.

۳.۴. عملگر انقلاب

بروز انقلاب باعث تغییرات ناگهانی در ویژگی‌های اجتماعی ـ سیاسی یک کشور می‌شود. در الگوریتم رقابت استعماری، انقلاب با جابه‌جایی تصادفی یک کشور مستعمره به یک موقعیت تصادفی جدید مدل‌سازی می‌شود. انقلاب از دیدگاه الگوریتمی باعث می‌شود که حرکت تکاملی از گیرکردن در دام بهینه‌ی موضعی نجات یابد که در بعضی موارد باعث بهبود موقعیت یک کشور شده و آن را به محدوده‌یی با وضعیت بهتر انتقال می‌دهد. عملگر انقلاب همانند عملگر جهش در الگوریتم ژنتیک عمل می‌کند. این عملگر موجب ایجاد تنوع در جواب‌های حاصل از سیاست جذب خواهد شد. عملگر انقلاب به‌منظور جلوگیری از همگرایی زودرس بعد از سیاست جذب اجرا می‌شود.

در الگوریتم پیشنهادی، در هر امپراطوری هر کدام از مستعمره‌ها با احتمال مشخصی انقلاب خواهند کرد. لذا یک عدد تصادفی بین صفر و ۱ تولید می‌کنیم، اگر کوچک‌تر یا مساوی احتمال انقلاب باشد سیاست انقلاب را اعمال می‌کنیم. برای این منظور دو روش اجرا می‌شود؛ در روش اول، با توجه به دسته‌بندی کارها، دو دسته‌ی تصادفی انتخاب کرده سپس دو کار به تصادف از دو دسته، انتخاب و جای دو کار را با هم جابه‌جا می‌کنیم به‌طوری که محدودیت ظرفیت دسته‌ها رعایت شود؛ در غیر این صورت مجدداً دو دسته به تصادف انتخاب می‌کنیم. بنابراین در این روش، انتخاب دسته‌ها و جابه‌جایی کارها هر دو به تصادف انجام می‌شود.[۲] در روش دیگر، یک کار به تصادف انتخاب می‌شود و در دسته‌یی دیگر قرار می‌گیرد. با استفاده از رویکرد الگوریتم Best Fit ابتدا ظرفیت باقی‌مانده‌ی دسته‌های شکل‌گرفته را به‌ترتیب صعودی مرتب می‌کنیم و در اولین دسته‌یی که ظرفیت باقی‌مانده‌ی آن بیشتر از اندازه‌ی کار انتخاب شده باشد، کار منتخب را دسته‌بندی می‌کنیم اگر چنین دسته‌یی موجود نباشد دسته‌ی جدیدی تشکیل می‌دهیم.

۴.۴. جابه‌جایی موقعیت استعمارگر و مستعمره

در حین حرکت مستعمرات به سمت کشور استعمارگر و اجرای سیاست انقلاب، ممکن است بعضی از این مستعمرات به موقعیت بهتری نسبت به کشور استعمارگر دست یابند. در واقع به نقاطی در تابع هزینه برسند که هزینه‌ی کم‌تری را نسبت به مقدار تابع هزینه در موقعیت استعمارگر تولید کند. در این حالت استعمارگر و مستعمره جای‌شان را با هم عوض کرده و الگوریتم با کشور استعمارگر در موقعیت جدید ادامه خواهد یافت. در ادامه، این کشور استعمارگر جدید است که به اعمال سیاست همگون‌سازی بر مستعمرات خود می‌پردازد.

۵.۴. محاسبه‌ی قدرت کل یک امپراطوری

قدرت کشور استعمارگر به اضافه‌ی درصدی از قدرت کل مستعمرات آن، قدرت کل یک امپراطوری است. بنابراین هزینه‌ی کل یک امپراطوری از طریق معادله‌ی ۱۳ محاسبه می‌شود:

$$TC_n = c_n + \xi\{mean(c_n^i)\} \qquad (۱۳)$$

که در آن، TC_n هزینه‌ی کل امپراطوری nام است. c_n چنان که پیش‌تر بیان شد هزینه‌ی استعمارگر nام است. $mean(c_n^i)$ نیز میانگین هزینه‌ی مستعمره‌های متعلق به امپراطوری nام است. ξ عددی مثبت است.

۶.۴. رقابت استعماری

هر امپراطوری که نتواند بر قدرت خود بیفزاید و قدرت رقابت خود را از دست بدهد در جریان رقابت‌های استعمارگری تدریجاً حذف خواهد شد. به‌عبارت دیگر به‌مرور زمان امپراطوری‌های ضعیف مستعمرات خود را از دست داده و امپراطوری‌های قوی‌تر این مستعمرات را تصاحب کرده و قدرت خود را افزایش می‌دهند. برای مدل کردن این واقعیت فرض می‌شود که امپراطوری در حال حذف، ضعیف‌ترین امپراطوری موجود است. لذا در هر تکرار از الگوریتم یک یا تعدادی از ضعیف‌ترین مستعمرات، ضعیف‌ترین امپراطوری را برداشته و برای تصاحب این مستعمرات، بین تمام امپراطوری‌های رقابت ایجاد می‌شود. مستعمرات مذکور لزوماً توسط قوی‌ترین امپراطوری تصاحب نخواهد شد، بلکه امپراطوری‌های قوی‌تر احتمال تصاحب بیشتری دارند. برای مدل‌سازی رقابت میان امپراطوری‌ها به‌منظور تصاحب این مستعمره، ابتدا از روی هزینه‌ی کل امپراطوری هزینه‌ی کل هنجارشده‌ی آن از طریق معادله‌ی ۱۴ تعیین می‌شود:

$$NTC_n = \exp(\frac{-TC_n}{\max TC_n}) \qquad i = ۱, ۲, ..., nEmp \qquad (۱۴)$$

که در آن NTC_n هزینه‌ی کل هنجارشده‌ی nامین امپراطوری، TC_n هزینه‌ی کل nامین امپراطوری، و $\max TC_n$ بیشترین هزینه‌ی کل در میان تمام امپراطوری‌هاست (هزینه‌ی کل ضعیف‌ترین امپراطوری). با داشتن هزینه‌ی کل هنجارشده، احتمال تصاحب مستعمره‌ی رقابت توسط هر امپراطوری از طریق معادله‌ی ۱۵ محاسبه

می‌شود. با داشتن احتمال تصاحب هر امپراطوری، مستعمره‌ی مذکور به امپراطوری حاصل از روش چرخه رولت تخصیص داده می‌شود.

$$pemp_n = \frac{NTC_n}{\sum_{i=1}^{nEmp} NTC_i} \quad (15)$$

۷.۴. حذف امپراطوری‌های ضعیف

همان‌گونه که بیان شد، در جریان رقابت‌های استعماری به تدریج امپراطوری‌های ضعیف سقوط کرده و مستعمرات‌شان به دست امپراطوری‌های قوی‌تر می‌افتد. در الگوریتم پیشنهادی، وقتی یک امپراطوری هیچ مستعمره‌یی نداشته باشد به‌عنوان مستعمره به یک امپراطوری دیگر اضافه می‌شود. انتخاب امپراطوری از طریق چرخه رولت صورت می‌پذیرد.

۸.۴. شرط توقف

الگوریتم رقابت استعماری در حالت کلی تا زمانی که تنها یک امپراطوری باقی مانده باشد ادامه می‌یابد. در الگوریتم پیشنهادی به‌منظور اطمینان از اتمام الگوریتم در زمانی منطقی و همچنین شرایط یکسان برای مقایسه با سایر الگوریتم‌های فراابتکاری در نظرگرفته شده، شرط توقف یک مقدار ثابت زمان پردازش است. این مقدار ثابت برابر است با $n * c * ۰/۲$ که n تعداد کارها و c تعداد مراحل مختلف کاری است. در چنین موقعیتی رقابت استعماری به پایان رسیده و الگوریتم متوقف می‌شود.

۹.۴. تنظیم پارامترهای الگوریتم رقابت استعماری

در این زیربخش رفتار الگوریتم پیشنهادی با پارامترهای مختلف مورد ارزیابی قرار می‌گیرد. ازجمله روش‌های توان‌مند آماری که برای تنظیم پارامترها مورد استفاده قرار می‌گیرد روش تاگوچی است. در روش طراحی آزمایشات تاگوچی با دو نوع فاکتور، فاکتورهای قابل کنترل (سیگنال) و فاکتورهای غیر قابل کنترل (نویز) مواجه می‌شویم. تغییرات ایجاد شده در خواص یک الگوریتم در حین آزمایش ناشی از تغییر فاکتورهای سیگنال است. اما در آزمایشات مجموعه‌یی از فاکتورها و پارامترهای غیر قابل کنترل وجود دارد که در طراحی مورد نظر لحاظ نشده است، گرچه بر خواص الگوریتم اثرگذارند. نسبت سیگنال به نویز (S/N) نشان‌دهنده‌ی حساسیت مشخصه‌ی عملکرد مورد بررسی به فاکتورهای نویز است. در هر آزمایش ما به دنبال بالاترین نسبت (S/N) در نتایج هستیم. مقدار بالای این نسبت نشان می‌دهد که اثر فاکتورهای سیگنال بیشتر از اثر فاکتورهای نویز است. در روش تاگوچی توابع هدف به سه دسته تقسیم می‌شوند. دسته‌ی اول برای مسائل با تابع هدف بیشینه‌سازی است که هرچه بیشتر باشد، بهتر است. دسته‌ی دوم برای مسائل با تابع هدف کمینه‌سازی است که هرچه کم‌تر باشد بهتر است و دسته‌ی سوم برای مسائلی است که در آنها رسیدن به یک مقدار اسمی مورد نظر است؛ هر چه به آن مقدار اسمی نزدیک‌تر باشد بهتر است. با توجه به این‌که تابع هدف مورد بررسی در این پژوهش از نوع کمینه‌سازی است به‌منظور تبدیل نتایج آزمایشات به نسبت (S/N) از معادله‌ی ۱۶ استفاده شده است.

$$\frac{S}{N} = -۱۰ \log(\frac{۱}{n}\sum_{i=1}^{n} y_i^۲) \quad (16)$$

y_i مقدار تابع هدف است. در این پژوهش، چهار پارامتر الگوریتم رقابت استعماری ــ شامل اندازه جمعیت ($nPop$)، تعداد امپراطوری‌ها ($nEmp$)، ضریبی از قدرت مستعمرات که در محاسبه‌ی قدرت کل امپراطوری به‌کار می‌رود ($Zeta$)، و احتمال انقلاب ($pRevo$) ــ باید تنظیم شود که برای هریک سه سطح در نظرگرفته شده است.

در جدول ۱ سطوح مختلف پارامترها نشان داده شده است (n نشان‌دهنده‌ی تعداد کارهاست). با توجه به تعداد پارامترها و سطوح انتخابی و مراجعه به ارائه‌های متعامد استاندارد تاگوچی، ۹ آزمایش برای تنظیم پارامترهای الگوریتم رقابت استعماری انتخاب و در جدول ۲ نشان داده شده است. برای تحلیل داده‌ها از نرم‌افزار مینی‌تب ۶ استفاده شده است.

مجموعه‌یی شامل ۵۴ مسئله، برای هر آزمایش در نظرگرفته می‌شود. تعداد کارها {۲۰, ۵۰, ۱۰۰}، تعداد مراحل مختلف کاری {۵, ۱۰} و تعداد ماشین موازی در هر مرحله {۲, ۳, ۴} در نظرگرفته شده است.[۱۴] لذا ۱۸ ترکیب به‌وجود می‌آید و برای هر ترکیب دو مسئله تولید می‌شود. زمان پردازش کارها در هر مرحله از توزیع یکنواخت در بازه ۱ تا ۹۹، اندازه کارها در بازه ۱ تا ۱۰، و در نهایت ظرفیت ماشین‌های هر مرحله در بازه ۲۰ تا ۳۰ تولید شده است.

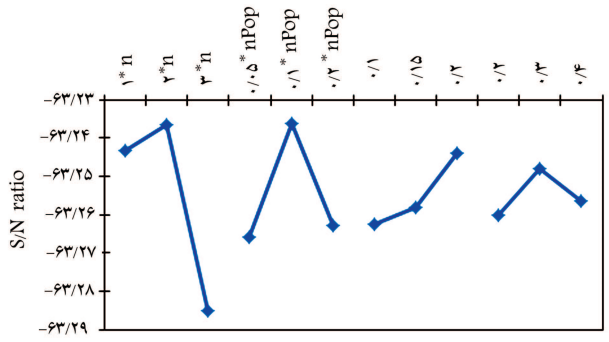
مسائل را با الگوریتم رقابت استعماری حل کرده و جواب‌ها را با توجه به

جدول ۱. پارامترها و سطوح آنها.

پارامترهای الگوریتم	شماره‌ی سطوح	سطوح پارامترها
اندازه جمعیت	۱	$۱ * n$
	۲	$۲ * n$
	۳	$۳ * n$
تعداد امپراطوری	۱	$۰/۰۵ * nPop$
	۲	$۰/۱ * nPop$
	۳	$۰/۲ * nPop$
ضریب قدرت مستعمرات	۱	۰/۱
	۲	۰/۱۵
	۳	۰/۲
احتمال انقلاب	۱	۰/۲
	۲	۰/۳
	۳	۰/۴

جدول ۲. ارائه‌های متعامد برای تنظیم پارامترهای الگوریتم.

آزمایش	اندازه‌ی جمعیت	تعداد امپراطوری	ضریب قدرت مستعمرات	احتمال انقلاب
۱	$۱ * n$	$۰/۰۵ * nPop$	۰/۱	۰/۲
۲	$۱ * n$	$۰/۱ * nPop$	۰/۱۵	۰/۳
۳	$۱ * n$	$۰/۲ * nPop$	۰/۲	۰/۴
۴	$۲ * n$	$۰/۰۵ * nPop$	۰/۱۵	۰/۴
۵	$۲ * n$	$۰/۱ * nPop$	۰/۲	۰/۲
۶	$۲ * n$	$۰/۲ * nPop$	۰/۱	۰/۳
۷	$۳ * n$	$۰/۰۵ * nPop$	۰/۲	۰/۳
۸	$۳ * n$	$۰/۱ * nPop$	۰/۱	۰/۴
۹	$۳ * n$	$۰/۲ * nPop$	۰/۱۵	۰/۲



شکل ۳. نمودار اثرات عمده‌ی نسبت سیگنال به نویز برای تنظیم پارامترها.

معادله‌ی ۱۶ به نسبت S/N تبدیل کرده و شکل ۳ را رسم می‌کنیم. با توجه به شکل اگر پارامتر اندازه جمعیت $n*۲$، تعداد امپراطوری $nPop*۰/۱$، ضریبی از قدرت مستعمرات در محاسبه‌ی قدرت کل امپراطوری ۰/۲ و احتمال انقلاب ۰/۳ باشد کیفیت حل الگوریتم رقابت استعماری بهتر خواهد بود.

۵. نتایج محاسباتی

از آزمایشات عددی برای ارزیابی عملکرد استفاده شده است. برای این منظور دو آزمایش مختلف طراحی می‌شود، آزمایش اول شامل تعدادی مسئله در اندازه کوچک است. مسائل توسط نرم‌افزار تخصصی بهینه‌سازی حل شده‌اند. آزمایش دوم شامل مسائلی با اندازه بزرگ‌تر است که عملکرد الگوریتم رقابت استعماری را با الگوریتم شبیه‌سازی تبرید[۲] و بهینه‌سازی اجتماع ذرات[۱۱] مقایسه می‌کند. برای انجام این آزمایش‌ها از CPLEX و Matlab۱۰ استفاده شده است و با رایانه‌ی پنج‌هسته‌یی با پردازش‌گر مرکزی ۲/۵ گیگاهرتز و حافظه‌ی اصلی چهار گیگابایت اجرا شده است.

مشخصات مسائل در اندازه کوچک شامل، تعداد کارها $\{۵, ۶, ۷, ۸, ۹, ۱۰\}$، تعداد مراحل مختلف کاری $\{۲, ۳\}$ و تعداد ماشین موازی در هر مرحله $\{۲\}$ در نظر گرفته شده است. لذا ۱۲ ترکیب از تعداد کارها، تعداد مراحل مختلف کاری، و تعداد ماشین موازی در هر مرحله به وجود می‌آید و دو مسئله برای هر ترکیب ایجاد شده که در مجموع ۲۴ مسئله تولید می‌شود. زمان پردازش کارها در هر مرحله به‌طور تصادفی از توزیع یکنواخت در بازه ۱۲ تا ۱۸ تولید می‌شود. اندازه کارها به‌طور تصادفی از توزیع یکنواخت در بازه ۱ تا ۵ تولید می‌شود. ظرفیت ماشین‌ها در هر مرحله به‌طور تصادفی از توزیع یکنواخت در بازه ۱۰ تا ۲۰ تولید می‌شود. بیشینه زمان اجازه داده شده برای محاسبات به نرم‌افزار ۱۵۰۰ ثانیه است. تعداد مسائل حل شده و میانگین زمان محاسباتی در جدول ۳ درج شده است.

چنان که مشاهده می‌شود ۲۱ مسئله از ۲۴ مسئله‌ی در نظرگرفته شده توسط مدل ریاضی در زمان محاسباتی ۱۵۰۰ ثانیه، حل شده است. بنابراین نتیجه می‌گیریم که مدل ریاضی به‌طور تقریبی قابلیت حل مسائل تا تعداد ۱۰ کار، دو مرحله‌ی کاری و دو ماشین موازی در هر مرحله را دارد.

۱.۵. بررسی الگوریتم‌ها در مسائل اندازه کوچک

ابتدا مسائلی را که با اندازه کوچک تولید شده‌اند توسط الگوریتم رقابت استعماری، شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات حل کرده و درصد انحراف نسبی آنها

جدول ۳. نتایج حاصل از حل مسائل اندازه کوچک با مدل ریاضی.

تعداد کارها	تعداد مراحل	مسئله	زمان (ثانیه)
۵	۲	۲	۰/۱۲
	۳	۲	۰/۱۵
۶	۲	۲	۰/۱۴
	۳	۲	۰/۱۹
۷	۲	۲	۱۸
	۳	۲	۱۰۰
۸	۲	۲	۱۰۳/۵
	۳	۲	۲۵۸
۹	۲	۲	۱۴۰
	۳	۱	۳۷۳
۱۰	۲	۲	۷۰۰
	۳	۰	۱۵۰۰

براساس معادله‌ی ۱۷ به‌دست می‌آید.

$$RPD = \frac{Alg_{sol} - Min_{sol}}{Min_{sol}} \times ۱۰۰ \qquad (۱۷)$$

Alg_{sol} مقدار تابع هدف به‌دست آمده از اجرای هر آزمایش در یک مسئله از الگوریتم مورد نظر است. Min_{sol} کم‌ترین مقدار تابع هدف به‌دست آمده برای مسئله است.

نتایج حاصل از مقایسه‌ی عملکرد این الگوریتم‌ها با جواب بهینه‌ی به‌دست آمده در جدول ۴ ارائه شده است. هر ردیف حاوی متوسط درصد انحراف نسبی دو مسئله است.

چنان که در جدول ۵ مشاهده می‌شود الگوریتم رقابت استعماری با میانگین درصد انحراف نسبی ۰/۳۷ عملکرد بهتری نسبت به دو الگوریتم دیگر در حل مسائل با اندازه کوچک دارد.

۲.۵. بررسی الگوریتم‌ها در مسائل اندازه بزرگ

در این زیربخش، عملکرد الگوریتم پیشنهادی رقابت استعماری با الگوریتم شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات موجود در ادبیات در حل مسائل با اندازه بزرگ مقایسه شده است. مشخصات مسائل در بخش تنظیم پارامتر ذکر شده است با این تفاوت که برای هر ترکیب ۱۰ مسئله به‌صورت تصادفی تولید می‌شود؛ بنابراین مجموعه‌یی شامل ۱۸۰ مسئله برای مقایسه‌ی تصادفی الگوریتم‌ها تولید شده است. تمامی مسائل با الگوریتم رقابت استعماری، شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات حل شده است و درصد انحراف نسبی آنها از طریق معادله‌ی ۱۶ به‌دست می‌آید (جدول ۵). هر ردیف نشان‌گر متوسط درصد انحراف نسبی ۱۰ مسئله است.

برای بررسی اختلاف بین میانگین الگوریتم‌ها، آزمون تحلیل واریانس یک‌طرفه روی مقادیر درصد انحراف نسبی اجرا شده است. با توجه به صفر بودن مقدار P عدم برابری میانگین الگوریتم‌ها را می‌توان نتیجه گرفت. تحلیل واریانس مربوطه، برای الگوریتم‌های ذکر شده در جدول ۶ ارائه شده است.

برای بررسی عملکرد الگوریتم‌ها از میانگین و فاصله‌ی اطمینان ۰/۹۵ روی

جدول ۴. میانگین درصد انحراف نسبی در مسائل با اندازه کوچک.

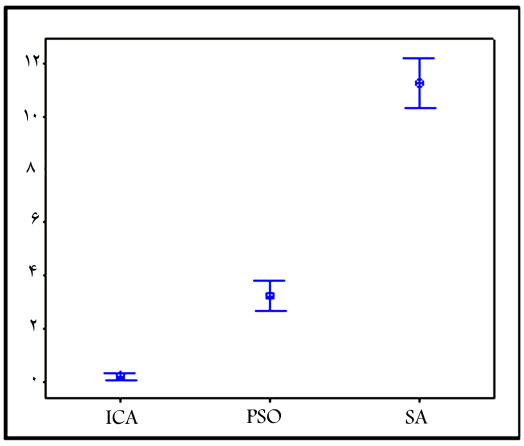
کار	مرحله	رقابت استعماری	شبیه‌سازی تبرید	بهینه‌سازی اجتماع ذرات
۵	۲	۰	۰	۰
	۳	۰	۰	۰
۶	۲	۰	۰	۰
	۳	۰	۱٫۹۶	۰
۷	۲	۰	۰	۰
	۳	۰	۵٫۲۶	۰
۸	۲	۰	۳٫۲۶	۱٫۰۸
	۳	۰	۱٫۶۳	۰
۹	۲	۰	۵٫۷۱	۶٫۸
	۳	۱٫۱۱	۸٫۵۱	۲٫۵۵
۱۰	۲	۰	۱۰٫۴۸	۱٫۱۱
	۳	۳٫۴۱	۰٫۸۱	۲٫۴۵
میانگین		۰٫۳۷	۳٫۱۳	۱٫۱۶

جدول ۵. نتایج به‌دست آمده از الگوریتم‌ها در اندازه‌های مختلف.

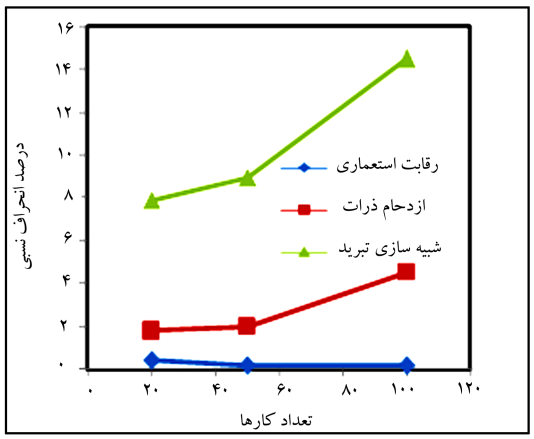
کار × مرحله	ماشین موازی در هر مرحله	رقابت استعماری	شبیه‌سازی تبرید	بهینه‌سازی اجتماع ذرات
۵ × ۲۰	۲	۱٫۲۰	۹٫۰۳	۰٫۶۷
۱۰ × ۲۰	۲	۰٫۲۷	۳٫۷۵	۱٫۲۴
۵ × ۵۰	۲	۰٫۳۳	۱۲٫۰۴	۱٫۵۳
۱۰ × ۵۰	۲	۰٫۰۴	۷٫۸۳	۱٫۰۵
۵ × ۱۰۰	۲	۰٫۱۴	۱۱٫۵۰	۱٫۲۳
۱۰ × ۱۰۰	۲	۰٫۱۹	۹٫۲۸	۰٫۷۴
۵ × ۲۰	۳	۰٫۷۹	۴٫۹۴	۰٫۸۵
۱۰ × ۲۰	۳	۰٫۰۱	۳٫۸۲	۱٫۸۶
۵ × ۵۰	۳	۰٫۰۳	۵٫۷۱	۱٫۳۴
۱۰ × ۵۰	۳	۰٫۱۲	۸٫۵۱	۱٫۷۹
۵ × ۱۰۰	۳	۰٫۳۵	۱۰٫۴۸	۰٫۷۴
۱۰ × ۱۰۰	۳	۰٫۱۳	۱۲٫۲۳	۱٫۹۴
۵ × ۲۰	۴	۰٫۵۱	۳٫۳۷	۱٫۲۹
۱۰ × ۲۰	۴	۰٫۱۸	۲٫۲۲	۲٫۲۱
۵ × ۵۰	۴	۰٫۰۹	۱۰٫۲۶	۲٫۵۸
۱۰ × ۵۰	۴	۰٫۴۸	۳٫۷۷	۰٫۷۱
۵ × ۱۰۰	۴	۰٫۰۵	۹٫۳۷	۱٫۲۶
۱۰ × ۱۰۰	۴	۰٫۱۹	۶٫۷۱	۱٫۰۶
میانگین		۰٫۲۹	۷٫۴۸	۱٫۳۴

جدول ۶. جدول تحلیل واریانس برای الگوریتم‌های رقابت استعماری، شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات.

منبع	درجه آزادی	مجموع مربعات	میانگین مربعات	آزمون F	P-Value
الگوریتم	۲	۵۲۱۲٫۴	۲۶۰۶٫۲	۲۸۶٫۴	۰٫۰۰
خطا	۵۳۷	۴۸۸۵٫۵۷	۹٫۱۰		
مجموع	۵۳۹	۱۰۰۹۸٫۱			



شکل ۴. نمایش میانگین و فاصله اطمینان ۰٫۹۵ برای الگوریتم‌ها.



شکل ۵. نمایش عملکرد الگوریتم‌ها به ازای تعداد کارهای مختلف.

مقادیر درصد انحراف نسبی به دست آمده در حل مسائل تصادفی استفاده شده است.[۱۵] چنان که در شکل ۴ مشاهده می‌شود، الگوریتم رقابت استعماری عملکرد بهتری نسبت به شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات برای حل مسئله‌ی زمان‌بندی جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی دارد.

برای تحلیل تأثیر مقادیر تعداد کار بر عملکرد الگوریتم‌ها، نتایج تحلیل شده است. در شکل ۵ میانگین انحراف نسبی به‌دست آمده از الگوریتم‌ها در مقادیر مختلف تعداد کار نشان داده شده است. الگوریتم رقابت استعماری به‌ازای افزایش تعداد کارها عملکرد بهتر و پایدارتری از خود ارائه می‌دهد. از سوی دیگر دو الگوریتم شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات به‌ازای افزایش تعداد کارها عملکرد ناپایداری ارائه می‌دهند.

۶. نتیجه‌گیری

این پژوهش به مطالعه‌ی یک سیستم تولیدی در محیط جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی اختصاص دارد. ماشین‌های پردازش دسته‌یی از قابلیت پردازش همزمان چندکار در یک دسته برخوردارند. در این حالت به‌جای پردازش تکی کارها تعدادی کار هم‌زمان در یک دسته پردازش می‌شود. پردازش کارها در این حالت ارزان‌تر و سریع‌تر از پردازش آنها به‌صورت تکی است.

ابتدا بحثی در مورد مدل‌سازی مسئله‌ی زمان‌بندی جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی، در قالب برنامه‌ریزی خطی عدد صحیح مختلط مطرح شد که نتایج به‌دست آمده عبارت است از:

- مدل ریاضی به‌عنوان یک روش حل برای به‌دست آوردن جواب بهینه مسائل در اندازه کوچک است.
- برای حل مسائل در اندازه‌های بزرگ‌تر الگوریتم فرابتکاری رقابت استعماری پیشنهاد شده است.
- الگوریتم رقابت استعماری نسبت به دو الگوریتم شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات برای حل مسئله‌ی زمان‌بندی جریان کارگاهی ترکیبی با ماشین‌های پردازش دسته‌یی، عملکرد بهتری دارد.
- الگوریتم رقابت استعماری به‌ازای افزایش تعداد کارها در مسئله‌ی زمان‌بندی عملکرد بهتر و پایدارتری از خود نشان می‌دهد. از سوی دیگر دو الگوریتم شبیه‌سازی تبرید و بهینه‌سازی اجتماع ذرات به‌ازای افزایش تعداد کارها عملکرد ناپایداری ارائه می‌دهند.

پانوشت‌ها

1. batch processing machines
2. printed circuit board
3. metal working
4. hybrid flowshop
5. imperialist competitive algorithm (ICA)
6. Roulete wheel selection

منابع (References)

1. Albers, S. and Brucker, P. "The Complexi-ty of one-machine batching problem", *Discrete Applied Mathematics,* **47**, pp: 87-107 (1993).
2. Manjeshwar, P., Damodaran, P. and Srihari, K. "Minimizing makespan in a flowshop with two batch-processing machines using simulated annealing", *Robot Comput Integr Manuf,* **25**, pp. 667-679 (2009).
3. Ahmadi, J.H., Ahamadi, R.H., Dasu, S. and Tang, C.S. "Batching and scheduling jobs on batch and discrete processors", *Operations Research,* **40**(4), pp. 750-763 (1992).
4. Damodaran, P. and Srihari, K. "Mixed integer formulation tominimize makespan in a flow shop with batch processing machines", *Mathematical and Computer Modelling,* **40**(13), pp. 1465-1472 (2004).
5. Liao, C.J. and Liao, L.M. "Improved MILP models for two-machine flowshop with batch processing time", *Mathematical and Computer Modelling,* **48**(7-8), pp. 1254-1264 (2008).
6. Muthuswamy, S., Mario, C., Maya, J. and Rojas-Santiago, M. "Minimizing makespan in a two-machine no-wait flow shop with batch processing machines", *International Journal of Advanced Manufacturing Technology,* **63**, pp. 281-290 (2012).
7. Muthuswamy, S., Mario, C., Maya, J. and Rojas-Santiago, M. "Minimizing makespan in a two-machine no-wait flow shop with batch processing machines", *International Journal of Advanced Manufacturing Technology,* **63**, pp. 281-290 (2012).
8. Liao, L.M. and Huang, C.G. "Tabu search heuristic for two-machineflowshop with batch processing machines", *Computers and Industrial Engineering*, **60**, pp. 426-432 (2011).
9. Lei, D.M. and Guo, X.P. "Variable neighborhood search for minimizing tardiness objectives on flow shop with batch processing machines", *International Journal of Production Research,* **49**, pp. 519-529 (2011).
10. Kashan, A.H. and Karimi, B. "An improved mixed integer linear formulation and lower bounds for minimizing makespan on a flow shop with batch processing machines", *International Journal of Advanced Manufacturing Technology,* **40**, pp. 582-594 (2009).
11. Damodaran, P., Rao, A.G. and Mestry, S. "Particle swarm optimization for scheduling batch processing machines in a permutation flowshop", *International Journal of Advanced Manufacturing Technology*, **64**(5-8), pp. 989-1000 (2013).
12. Amin-Naseri, M.R. and Beheshti-Nia, M.A. "Hybrid flow shop scheduling with parallel batching", *International Journal Production Economics,* **117**, pp. 185-196 (2009).
13. Atashpaz-Gargari, E. and Lucas, C. "Imper-ialist competitive algorithm: An algorithm for optimization inspired by imperialistic comp-etition", *IEEE Congress,* pp. 4661-70 (2007).
14. Bozejko, W., Pempera, J. and Smutnicki, C. "Parallel tabu search algorithm for the hybrid flow shop problem", *Computers and Industrial Engineering.* **65**, pp. 466-474 (2013).
15. Attar, S.F., Mohammadi, R. and Tavakkoli-Moghaddam, M. "A novel imperialist competitive algorithm to solve flexible flow shop scheduling problem in order to minimize maximum completion time", *International Journal of Computer Applications*, **28**, pp. 0975-8887 (2011).