فصلنامه علمی – پژوهشی مطالعات مدیریت صنعتی – سال چهاردهم، شماره ۴۳، زمستان ۹۵
صفحات ۵۳-۷۷

زمان‌بندی جریان کاری ترکیبی با وجود کارهای بدون انتظار: مدل ریاضی و الگوریتم حل

بهمن نادری*

تاریخ دریافت : ۹۴/۹/۱۰
تاریخ پذیرش: ۹۵/۶/۹

چکیده

در این مقاله، مسئله زمانبندی جریان کاری ترکیبی با ماشین‌های موازی در هر یک از ایستگاه های کاری مورد بررسی قرار می‌گیرد. در این مسئله کارها به دو گروه تقسیم شده‌اند: کارهایی که باید به صورت بدون انتظار زمانبندی شوند و کارهای معمولی. کارهای بدون انتظار کارهایی محسوب می شوند که باید بین پردازش عملیات‌های آن کار نباید هیچ فاصله زمانی وجود داشته باشد. در این مقاله، پس از تشریح جوانب مختلف مسئله، دو مدل ریاضی قالب برنامه‌ریزی عددصحیح مختلط است. با استفاده از نرم‌افزارهای تجاری تخصصی تحقیق در عملیات مدل ریاضی حل و نتایج عملکردی آن تحلیل و مقایسه می‌شود. مدل‌های ریاضی تنها قادر به حل مسایل با اندازه کوچک هستند. سپس برای حل مسئله در اندازه‌های واقعی، دو الگوریتم فراابتکاری شامل الگوریتم رقابت استعماری و شبیه‌سازی تبرید طراحی می‌شود. یک مجموعه مثال آزمایشگاهی تولید و عملکرد الگوریتم‌ها با یکدیگر مقایسه می‌شود. الگوریتم رقابت استعماری در مقایسه با الگوریتم دیگر عملکرد بهتری دارد.

واژگان کلیدی: جریان کاری ترکیبی، زمانبندی بدون انتظار، مدل ریاضی برانمه ریزی عددصحیح مختلط، شبیه سازی تبرید، الگوریتم رقابت استعماری.

* هیات علمی، گروه مهندسی صنایع، دانشگاه خوارزمی تهران bahman.naderi@aut.ac.ir

مقدمه

زمانبندی، تعیین توالی کارها جهت تخصیص به منابع تولید است که برای رسیدن به اهدافی مانند بهره‌برداری بهتر از منابع در دسترس، پاسخگویی سریع به تقاضای مشتریان و کمینه کردن زمان تکمیل کل کارها انجام می‌شود. با توجه به تعداد عملیاتی که برای پردازش یک کار مورد نیاز است و تعداد ماشین‌های موجود برای انجام هر عملیات، مسائل زمانبندی را می‌توان تفکیک کرد. محیط جریان کارگاهی شامل چند مرحله متوالی است که در هر مرحله عملیات خاصی بر روی کارها انجام می‌شود. کارها جهت پردازش از تمامی مراحل به ترتیب باید عبور کنند. با توجه به شرایط رقابتی صنایع، بسیاری از سیستم‌های تولیدی کلاسیک به سمت محیط‌های جدیدتری مانند جریان کارگاهی ترکیبی، تغییر جهت می‌دهند که این سیستم تولیدی، ترکیبی از جریان کارگاهی و ماشین‌های موازی است. سیستم جریان کارگاهی، به منظور افزایش ظرفیت خود در مراحلی که گلوگاه است از ماشین های 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.

مسائل زمانبندی تولید بدون تاخیر از جمله پرکاربردترین فضاهای تولیدی محسوب می‌شوند. صنایع ذوب آهن یکی از دهها محیط واقعی محسوب می‌شوند که کارها باید به طور پیوسته روی پردازشگرها اجرا شوند. علاقمندان برای مطالعه دقیق‌تر کاربردهای دیگر مسائل به مقالات هال و سریسکانداراجاه[1] (۱۹۹۶) و گویال و سریسکانداراجاه[2] (۱۹۹۸) ارجاع داده می‌شوند. پیچیدگی مسائل زمانبندی بدون تاخیر در (گیارو[3]، ۲۰۰۱) به طور جامع بحث می‌شود. راک[4] (۱۹۸۴) نشان می‌دهد که مسئله جریان کارگاهی بدون تاخیر یک مسئله سخت یا اصطلاحا NP-hard است.

1- Hall and Sriskandarajah
2- Goyal and Sriskandarajah
3- Giaro
4- Rock

راجنران[1] (۱۹۹۴) نیز الگوریتم‌های ابتکاری را بر اساس مفهوم همایسگی جایگذاری[2] برای این مسئله ارائه می‌کند. مسئله‌ای مشابه نیز توسط چن و نپالی[3] (۱۹۹۶) مورد بررسی قرار می‌گیرد. ایشان برای حل این مسئله الگوریتم فوق‌ابتکاری از جنس الگوریتم ژنتیک[4] ارائه می‌کنند. الدوواسن و الله‌وردی[5] (۲۰۰۴) یک الگوریتم ژنتیک به همراه چند رویه بهبود بر اساس الگوریتم شبیه سازی تبرید[6] ارائه می‌کنند.

الگوریتم ترکیبی الگوریتم ژنتیک و الگوریتم شبیه سازی تبرید نیز توسط اسچاستر و فرامینن[7] (۲۰۰۳) ارائه می‌شود. همین‌طور ایشان در همان مقاله الگوریتم جستجوی چند همسایگی[8] نیز توسعه می‌دهند. فینک و وب[9] (۲۰۰۳) نیز برای مسئله‌ای مشابه یک الگوریتم شبیه‌سازی تبرید و یک الگوریتم جستجوی تابو ارائه می‌کنند. چنگ و همکاران[10] (۲۰۰۷) نیز الگوریتم ژنتیکی را برای مسئله جریان کارگاهی بدون تاخیر با هدف مینیمم کردن زمان تکمیل کل کارها توسعه می‌دهند. پن[11] و همکاران (۲۰۰۸) یک الگوریتم حریص تکرار شونده[12] را ارائه کردند. در مقاله‌ای دیگر رویز و الله‌وردی[13] (۲۰۰۷) مسئله جریان کارگاهی بدون تاخیر را با در نظر گرفتن زمان‌های آماده‌سازی را برای مینیمم سازی زمان تکمیل کا کارها مطالعه کرده و یک الگوریتم جستجوی همسایگی تکرارشونده[14] را توسعه می‌دهند.

1- Rajendran
2- Insertion neighborhood
3- Chen and Neppalli
4- Genetic algorithm
5- Aldowaisan and Allahverdi
6- Simulated annealing
7- Schuster and Framinan
8- Variable neighborhood search
9- Fink and Voß
10- Chang et al.
11- Pan et al.
12- Iterated greedy algorithm
13- Ruiz and Allahverdi
14- Iterated local search

اخیرا، ناگانو[1] و همکاران (۲۰۱۴) مسئله جریان کارگاهی بدون تاخیر را مطالعه و یک الگوریتم تکاملی به نام جستجوی طبقه‌بندی شده ارائه نمودند. نادری و همکاران[2] (۲۰۱۴) یک الگوریتم جستجوی شکار و همچنین لاها و ساپکال[3] (۲۰۱۴) یک الگوریتم ابتکاری برای حل همین مسئله ارایه کردند.

توجه به مطالعات انجام شده، در تمامی مقالات در زمینه زمانبندی بدون تاخیر، فرض می‌شود که تمامی کارها چنین ویژگی دارند. در صورتی که ممکن است در یک کارگاه بعضی از کارها بدون تاخیر باشند و بعضی دیگر کارهای معمولی محسوب شوند (بدین معنی که می‌تواند بین عملیات‌های آن انتظار یا تاخیر نیز وجد داشته باشد. هدف این مقاله ارائه بررسی مسئله جریان کارگاهی مختلط با وجود دو گروه کار بدون تاخیر و معمولی است. در این مقاله، پس از تشریح جوانب مختلف مسئله، مدل ریاضی آن در قالب یک مدل برنامه‌ریزی ریاضی عددصحیح مختلط خطی توسعه داده می‌شود. با استفاده از نرم‌افزارهای تجاری تخصصی تحقیق در عملیات مدل ریاضی حل و نتایج عملکردی آن تحلیل می‌شود. سپس برای حل مسئله در اندازه‌های واقعی، دو الگوریتم فرابتکاری شامل الگوریتم ژنتیک و شبیه‌سازی تبرید طراحی می‌شود. یک مجموعه مثال آزمایشگاهی تولید و عملکرد الگوریتم‌ها با یکدیگر مقایسه می‌شود. الگوریتم ژنتیک در مقایسه با الگوریتم دیگر عملکرد بهتری دارد.

تعریف مساله و مدل ریاضی

مساله زمانبندی مورد بررسی در این پژوهش شامل ویژگی‌های زیر است. N کار که ابتدای افق زمان‌بندی در دسترس هستند و روی g مرحله باید پردازش شوند. در هر مرحله l، تعداد m_l ماشین به صورت موازی وجود دارد. تمامی کارها از تمامی مراحل باید عبور کنند و فقط روی یک ماشین در هر مرحله پردازش شوند. مسیر پردازش برای همه کارها یکسان است.

1- Nagano et al.

2- Naderi etal.

3- Laha and Sapkal

امکان قطع شدن عملیات وجود ندارد. ماشین‌های موازی در هر مرحله یکسان است. دو مجموع کار وجود دارد: مجموعه کارهای بدون تاخیر و معمولی.

در ادامه، دو مدل برنامه‌ریزی خطی عدد صحیح مختلط برای مساله مورد بررسی، ارائه می‌شود. ابتدا پارامترها و نمادهای استفاده شده در مدل‌ها توضیح داده شده است.

n	تعداد کارها
g	تعداد ایستگاه‌ها
k, j	شمارنده کارها $\{1, 2, \dots, n\}$
i	شمارنده ایستگاه‌ها $\{1, 2, \dots, g\}$
m_i	تعداد ماشین‌ها در ایستگاه i
l	شمارنده ماشین‌ها در ایستگاه i
$p_{j,i}$	زمان پردازش کار j در ایستگاه i
N_j	مجموعه کارهای بدون تاخیر
O_j	مجموعه کارهای معمولی
M	یک عدد بزرگ مثبت

اولین مدل پیشنهادی مبتنی بر یافتن توالی نسبی است. بدین معنی که متغیرهای باینری استفاده شده توالی نسبی کارهای مختلف روی ماشین‌ها را نشان می‌دهند. متغیرهای تصمیم این مدل شامل متغیرهای زیر است:

$X_{j,k,i}$ متغیر باینری که وقتی کار j بعد از کار k پردازش شود، عدد ۱ و در غیر این صورت عدد ۰ را می‌پذیرد جایی که داریم $k > j$

$Y_{j,i,l}$ متغیر باینری که وقتی کار j در ایستگاه توسط ماشین پردازش شود، عدد ۱ و در غیر این صورت عدد ۰ را می‌پذیرد.

$C_{j,i}$ متغیر پیوسته برای تعیین زمان تکمیل کار j روی ایستگاه i

مدل ریاضی ۱، مسئله جریان کارگاهی ترکیبی با وجود کارهای بدون تاخیر با هدف مینیمم کردن میکسپن را به صورت زیر فرمول‌بندی می‌کند:

$$Minimize\ C_{max}$$
$$Subject\ to:$$

$$\sum_{l=1}^{m_i} Y_{j,i,l} = 1 \qquad \forall_{i,j} \qquad (1)$$
$$C_{j,i} = C_{j,i-1} + p_{j,i} \qquad \forall_{i,j\in N_j} \qquad (2)$$
$$C_{j,i} \geq C_{j,i-1} + p_{j,i} \qquad \forall_{i,j\in O_j} \qquad (3)$$
$$C_{j,i} \geq C_{k,i} + p_{j,i} - M \cdot (3 - X_{j,i,k} - Y_{j,i,l} - Y_{k,i,l}) \qquad \forall_{i,l,j<k} \qquad (4)$$
$$C_{k,i} \geq C_{j,i} + p_{k,i} - M \cdot X_{j,i,k} - M \cdot (2 - Y_{j,i,l} - Y_{k,i,l}) \qquad \forall_{i,l,j<k} \qquad (5)$$
$$C_{max} \geq C_{j,m} \qquad \forall_{j} \qquad (6)$$
$$C_{j,i} \geq 0 \qquad \forall_{j,i} \qquad (7)$$
$$X_{j,i,k} \in \{0,1\} \qquad \forall_{i,j<k} \qquad (8)$$
$$Y_{j,i,l} \in \{0,1\} \qquad \forall_{i,l,j} \qquad (9)$$

مجموعه محدودیت (۱) مشخص می‌کند که هر کار در هر ایستگاه توسط کدام ماشین پردازش شود. مجموعه محدودیت (۲) مربوط به کارهای بدون تاخیر است که باید محدودیت بدون تاخیر آنها رعایت شود. مجموعه محدودیت (۳) اشاره دارد که یک کار توسط دو ایستگاه نمی‌تواند همزمان پردازش شود. مجموعه محدودیت (۴) و (۵) مشخص می‌کند که یک ماشین همزمان قادر به پردازش بیش از یک کار نیست. مجموعه محدودیت (۶) مقدار تابع هدف (میکسپن) را محاسبه می‌کند. مجموعه محدودیت (۷)، (۸) و (۹) نیز متغیرهای تصمیم را تعریف می‌کنند.

مدل دوم همچنان بر اساس متغیرهای توالی محور است با این تفاوت که مفهوم بلافاصله نیز در تعریف متغیر تصمیم استفاده شده است. متغیرهای مدل در زیر آورده شده است.

$X_{j,k,i}$ متغیر باینری که وقتی کار j بلافاصله بعد از کار k پردازش شود، عدد ۱ و در غیر این صورت عدد ۰ را می‌پذیرد جایی که داریم $k = j$

$S_{j,i}$ متغیر پیوسته برای تعیین زمان شروع کار jروی ایستگاه i

مدل ریاضی دوم به قرار زیر است:

$$Minimize\ C_{max}$$
$$Subject\ to:$$

$$\sum_{k\neq j}\sum_{l=1}^{m_i} X_{j,i,k,l} = 1 \qquad \forall_{i,j} \qquad (10)$$
$$\sum_{j,k\neq j}\sum_{l=1}^{m_i} X_{j,i,k,l} \leq 1 \qquad \forall_{i,k} \qquad (11)$$
$$\sum_{j} X_{j,i,0,l} = 1 \qquad \forall_{i,l} \qquad (12)$$
$$\sum_{j,k\neq j} X_{j,i,k,l} \leq \sum_{j,k\neq j} X_{k,i,j,l} \qquad \forall_{i,l,k} \qquad (13)$$
$$\sum_{l=1}^{m_i}\left(X_{j,i,k,l} + X_{k,i,j,l}\right) \leq 1 \qquad \forall_{i,j\neq k} \qquad (14)$$
$$S_{j,i+1} = S_{j,i} + p_{j,i} \qquad \forall_{i<m, j\in N_j} \qquad (15)$$
$$S_{j,i+1} \geq S_{j,i} + p_{j,i} \qquad \forall_{i<m, j\in O_j} \qquad (16)$$
$$S_{j,i} \geq S_{k,i} + p_{k,i} - M\cdot\left(1 - \sum_{l=1}^{m_i} X_{j,i,k,l}\right) \qquad \forall_{i,k\neq j} \qquad (17)$$
$$C_{max} \geq S_{j,m} + p_{j,m} \qquad \forall_{j} \qquad (18)$$
$$S_{j,i} \geq 0 \qquad \forall_{j,i} \qquad (19)$$
$$X_{j,i,k,l} \in \{0,1\} \qquad \forall_{i,l,j\neq k} \qquad (20)$$

مجموعه محدودیت‌های (۱۰) مشخص می‌کند که هر کار در هر ایستگاه توسط کدام ماشین پردازش شود و کار قبلی آن چیست. مجموعه محدودیت‌های (۱۱) اشاره دارد که هر کار روی هر کار می‌تواند حداکثر یک کار بعدی داشته باشد در حالی که مجموعه محدودیت‌های (۱۲) اشاره دارد که در توالی، کار مجازی 0 حتما یک کار بعدی دارد. مجموعه محدودیت (۱۳) مشخص می‌کند که هر کار توسط تنها یک ماشین در هر ایستگاه پردازش شود. مجموعه محدودیت (۱۴) از ایجاد جواب‌های غیر موجه بدین صورت که یک کار همزمان کار بعدی و قبلی کار دیگری باشد جلوگیری می‌کند. مجموع محدودیت (۱۵) مشخص می‌کند محدودیت بدون تاخیر را برای کارها مربوطه رعایت می‌کند در حالی که مجموعه محدودیت (۱۶) نشان می‌دهد که یک کار حداکثر همزمان توسط یک ایستگاه پردازش شود. مجموعه محدودیت (۱۷) نشان می‌دهد که یک ماشین حداکثر همزمان می‌تواند یک کار را پردازش کند. مجموعه محدودیت (۱۸) مقدار میکسپن را محاسبه می‌کند. متغیرهای مدل توسط دو مجموعه محدودیت (۱۹) و (۲۰) تعریف می‌شوند.

الگوریتم‌های حل پیشنهادی

از آنجایی که مسئله مورد بررسی از جمله مسائل NP-hard محسوب می‌شود، برای حل آن دو الگوریتم رقابت استعماری و شبیه‌سازی تبرید توسعه داده می‌شود.

الگوریتم رقابت استعماری

الگوریتم رقابت استعماری، یک الگوریتم فراابتکاری مبتنی بر جمعیت است. این الگوریتم برای اولین بار توسط آتش‌پز و لوکاس[1] (۲۰۰۷) ارائه شد. از جمله علمکردهای موفق دیگر این الگوریتم می‌توان به مقاله عطار و همکاران[2] (۲۰۱۱) اشاره کرد. معمولا الگوریتم‌های فراابتکاری، از یک پدیده طبیعی الهام می‌گیرند. این الگوریتم علاوه بر توجه به تکامل زیستی انسان و سایر موجودات، به تکامل اجتماعی و تاریخی او به عنوان پیچیده‌ترین و موفق‌ترین حالت تکامل، توجه می‌کند. شبه کد الگوریتم رقابت استعماری در شکل ۱ نشان داده شده است.

رویه الگوریتم رقابت استعماری
مکانیزم راه‌اندازی
حلقه: اگر شرایط توقف فرانرسیده است ادامه دهید
عملگر جذب
عملگر انقلاب
عملگر قدرت امپراطوری،
عملگر رقابت استعماری و حذف امپراطوری
پایان حلقه

شکل ۱. رویه کلی الگوریتم رقابت استعماری

1- Atashpaz and Lucas
2- Attar et al.

مرحله راه‌امدازی اولیه و شرایط توقف

اولین مرحله الگوریتم شامل نحوه نمایش جواب، تولید جمعیت اولیه و شکل‌دهی امپراطوری-های اولیه است. نحوه نمایش جواب به صورت یک بردار با n مولفه است که n نشان‌دهنده تعداد کارها است. به عبارتی یک ترتیب با اعداد صحیح 1 تا n یک جواب از مسئله محسوب می‌شود. کارها به ترتیب از چپ به راست انتخاب و زمانبندی می‌شود. برای زمانبندی، کار انتخاب شده به اولین ماسین در دسترس تخصیص می‌یابد. برای مرحله دو تا آخرین مرحله، کارها بر اساس زمان تکمیل مرحله قبل مرتب می‌شوند. جمعیت اولیه الگوریتم (تعداد کشورها) رقابت استعماری به صورت تصادفی تولید می‌شود. شرط توقف یک مقدار ثابت زمان پردازش است. این مقدار ثابت برابر است با ng.

تابع هزینه هر یک از کشورها محاسبه و به تعداد امپراطوری، از بهترین اعضای این جمعیت که دارای کمترین مقدار تابع هزینه هستند به عنوان استعمارگر انتخاب می‌شوند. جمعیت باقیمانده به عنوان کشورهای مستعمره هستند که هرکدام به یک امپراطوری تعلق دارند. برای تقسیم مستعمرات اولیه بین استعمارگرها از انتخاب چرخه رولت استفاده شده است. برای انجام این کار با داشتن هزینه همه استعمارگرها، هزینه نرمالیزه آن‌ها بر اساس معادله زیر محاسبه می‌گردد.

$$C_n = e^{(-c_n / \max c_i)}, \quad i = 1,2,\dots,nEmp$$

c_n هزینه nامین استعمارگر است، C_n هزینه نرمالیزه شده‌ی nامین استعمارگر است و $\max c_i$ بیشینه هزینه در بین تمام استعمارگرها است. $nEmp$ تعداد امپراطوری‌ها است. با داشتن هزینه نرمالیزه، قدرت نسبی نرمالیزه هر استعمارگر بر اساس معادله زیر محاسبه می‌گردد و مبنایی برای تقسیم کشورهای مستعمره بین استعمارگرها است.

$$P_n = \frac{C_n}{\sum_{i=1}^{nEmp} C_i}, \quad \sum_{i=1}^{nEmp} P_i = 1$$

روش چرخه رولت 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 یک تولید می‌شود. انتخاب این بازه به این دلیل است که مجموع مقادیر احتمال انتخاب، همیشه برابر با یک خواهد بود. از مقایسه عدد تصادفی با بازه چرخه رولت، شماره استعمارگر متناظر با عدد تصادفی مشخص می‌گردد. از آن‌جایی 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 همه کشورهای مستعمره به استعمارگرها تخصیص داده می‌شوند و با داشتن حالت اولیه تمام امپراطوری‌ها، الگوریتم رقابت استعماری شروع می‌شود. روند تکامل در یک حلقه قرار دارد که تا برآورده شدن شرط توقف، ادامه می‌یابد.

عملگر جذب و انقلاب

در هر امپراطوری، کشور استعمارگر به منظور افزایش نفوذ خود سعی می‌کند تعداد مستعمره‌هایش را افزایش دهد. لذا در هر امپراطوری کشورهای مستعمره شروع به حرکت به سمت استعمارگر مربوطه می‌نمایند. 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 تا کشور مستعمره را در راستای ابعاد مختلف اجتماعی سیاسی به خود نزدیک کند. در الگوریتم ارائه شده سیاست جذب به صورت زیر تعریف شده است:

برای تغییر در مستعمره بر اساس امپراطور، دو نقطه در مستعمره انتخاب قبل و بعد از این دو نقطه را حذف می‌کنیم. سپس کارهایی که حذف شده‌اند را بر اساس ترتیب نسبی آنها در استعمارگر به جواب اضافه می‌کنیم.

برای درک بهتر رویه جذب، یک مثال در ادامه توضیح داده می‌شود. یک مسئله با ۶ کار را در نظر بگیرید. فرض کنید امپراطور و مستعمره به صورت ذیل هستند.

امپراطور	۵	۴	۳	۶	۱	۲
مستعمره	۳	۲	۶	۵	۱	۴

اگر دو نقطه به ترتیب ۳ تا ۵ باشند. جواب جدید ایجاد شده، به صورت زیر خواهد بود:

مستعمره	۴	۲	۶	۵	۳	۱

بروز انقلاب، باعث تغییرات ناگهانی در ویژگی‌های اجتماعی سیاسی یک کشور است. در الگوریتم رقابت استعماری، انقلاب با جابه‌جایی تصادفی یک کشور مستعمره به یک موقعیت تصادفی جدید مدل‌سازی می‌شود. انقلاب از دیدگاه الگوریتمی باعث می شود که حرکت تکاملی از گیر کردن در دام بهینه موضعی نجات یابد که در بعضی موارد باعث بهبود موقعیت یک کشور شده و آن را به یک محدوده‌ای که وضعیت بهتری دارد، انتقال می‌دهد. در الگوریتم پیشنهاد شده، در هر امپراطوری هر کدام از مستعمره‌ها با احتمال مشخص شده‌ای، انقلاب خواهند کرد. در این عملگر، دو کار به تصادف انتخاب و جای دو کار را با هم جا به جا می‌کنیم.

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 این حالت استعمارگر و مستعمره جای خود را با همدیگر عوض کرده و الگوریتم با کشور استعمارگر در موقعیت جدید ادامه خواهد یافت. در ادامه، این کشور استعمارگر جدید است که شروع به اعمال سیاست همگون‌سازی بر مستعمرات خود می‌نماید.

قدرت یک امپراطوری، رقابت استعماری و حذف امپراطوری‌های ضعیف

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

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

TC_n، هزینه‌ی کل امپراطوری nام است. c_n همان‌طور که قبلا" ذکر شد هزینه استعمارگر nام است. $mean(c_n^i)$، میانگین هزینه‌ی مستعمره‌های متعلق به امپراطوری nام است. ξ عددی مثبت است.

هر امپراطوری که نتواند بر قدرت خود بیفزاید و قدرت رقابت خود را از دست بدهد در جریان رقابت‌های استعمارگری حذف خواهد شد. این حذف شدن به صورت تدریجی صورت می‌پذیرد. به عبارت دیگر به مرور 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 رقابتی در میان کلیه‌ی امپراطوری‌ها ایجاد می‌شود. مستعمرات مذکور لزوما توسط قوی‌ترین امپراطوری تصاحب نخواهد شد. بلکه امپراطوری‌های قویتر احتمال تصاحب بیشتری دارند. برای مدل-سازی رقابت میان امپراطوری‌ها برای تصاحب این مستعمره، ابتدا از روی هزینه کل امپراطوری هزینه کل نرمالیزه شده آن از طریق معادله زیر تعیین می‌گردد:

$$NTC_n = e^{(-TC_n/\max TC_i)}$$

NTC_n هزینه کل نرمالیزه شده nامین امپراطوری، TC_n هزینه کل nامین امپراطوری و $\max TC_i$ بیشترین هزینه‌ی کل در میان تمام امپراطوری‌ها است (هزینه‌ی کل ضعیف‌ترین امپراطوری). با داشتن هزینه کل نرمالیزه شده، احتمال تصاحب مستعمره رقابت توسط هر امپراطوری از طریق معادله زیر محاسبه می‌شود:

$$pemp_n = NTC_n / \sum_{i=1}^{nEmp} NTC_i$$

با داشتن احتمال تصاحب هر امپراطوری، مستعمره‌ی مذکور به امپراطوری که از روش چرخه رولت به‌دست آمده است تخصیص داده می‌شود.

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 مستعمره‌ای نداشته باشد به عنوان مستعمره به یک امپراطوری دیگر اضافه می‌شود. انتخاب امپراطوری از طریق چرخه رولت صورت می‌پذیرد.

الگوریتم شبیه‌سازی تبرید

جهت ساخت کریستال با کیفیت و خواص عالی بدین صورت زیر عمل می‌شود: ابتدا ماده جامد را بایستی تا یک دمای بسیار بالا حرارت داد (این دما باید به گونه‌ای انتخاب شود که اتم‌های ماده جامد بتوانند آزادانه حرکت کنند). پس از رسیدن دمای ماده به درجه حرارت مورد نظر، دمای ماده به صورت بسیار کند و تدریجی کاهش می‌یابد. در هر دمای ایجاد شده اتم‌ها تنها به اندازه ای می‌توانند جا به جا شوند که بیشترین پایداری را ایجاد کنند. به عبارت دیگر اگر ماده با کندی بیشتری سرد شود اتم‌ها قادر به آزادسازی انرژی بیشتر و قرارگیری در جهت بیشترین پایداری را خواهند داشت. در هر صورت سرد کردن سریع باعث می‌شود که ذرات فرصت پایین آمدن انرژی و در کنار هم قرار گرفتنشان کافی نباشد، لذا خیلی شکننده می‌شوند. با این حال سرد کردن ملایم ذرات، موجب می‌شود که ذرات فرصت یافته تا سطح انرژی خود را به حداقل رسانده و در کنار هم آرایش مناسبی داشته باشند. این فرآیند آهسته سرد کردن برای رسیدن به خواص عالی را تبرید گویند. این فرآیند توسط روش‌های بهینه سازی الگو برداری شد و باعث به وجود آمدن الگوریتم شبیه‌سازی تبرید گردید.

شبیه‌سازی تبرید از فرآیند تبرید در جامدات الهام می‌گیرد. تبرید فرآیندی است برای بهبود کیفیت خواص جامدات. ایده شبیه‌سازی تبرید که توسط متروپولیس[1] ارائه شده است.

1- Metropolis

الگوریتم مورد نظر مواد را به صورت سیستمی از ذرات شبیه‌سازی می‌کند. فرآیند سرد شدن به صورت کاهش تدریجی دمای [1]سیستم شبیه‌سازی می‌شود تا این که سیستم به یک حالت پایدار همگرا شود. بعدها، کرک پاتریک[2] و همکارانش ایده الگوریتم متروپولیس را مد نظر قرار داده و آن را در مسایل بهینه‌سازی ترکیباتی و برخی دیگر از مسایل به کار بردند.

شبیه‌سازی تبرید یک الگوریتم جستجوی محلی یا یک روش فوق ابتکاری می‌باشد که قادر است خود را از دام بهینه موضعی رها کند. راحتی بکارگیری، همگرایی و استفاده از حرکات خاصی جهت دوری از قرارگیری در دام بهینه موضعی از جمله خصوصیاتی هستند که باعث شده‌اند تا این روش در دو دهه اخیر مورد توجه قرار گیرد. نامگذاری این روش به نام شبیه‌سازی تبرید بیشتر به دلیل قیاس آن با فرآیند تبرید فیزیکی در جامدات است که در آن یک ماده جامد تا یک دمای بالا حرارت داده شده و سپس به تدریج سرد می‌شود تا این که بتواند به پایدارترین شبکه کریستالی خود دست یابد (حداقل انرژی ممکن را داشته باشد). بدین ترتیب یک جامد با کیفیت بالا و عاری از نقص به دست می‌آید. شبیه‌سازی تبرید ارتباطی بین این رفتار ترمودینامیکی و فرآیند جستجو برای یافتن نقاط بهینه برقرار می‌کند. می‌توان این گونه بیان کرد که روش شبیه‌سازی تبرید، حالت توسعه یافته روشی است که در بهینه سازی تحت عنوان تپه‌نوردی شناخته می‌شود رویه کلی الگوریتم شبیه‌سازی تبرید در شکل ۲ نشان داده شده است.

رویه الگوریتم شبیه‌سازی تبرید
مکانیزم راه‌اندازی
حلقه: اگر شرایط توقف فرانرسیده است ادامه دهید
عملگر حرکت
عملگر پذیرش
مکانیزم کاهش دما
پایان حلقه

شکل ۲. رویه کلی الگوریتم شبیه‌سازی تبرید

1- Gradual temperature decrease
2 -Kirkpatrick

یک مزیت مهم روش شبیه‌سازی تبرید نسبت به روش‌های دیگر، قابلیت این روش در جلوگیری از گیر افتادن در بهینه موضعی می‌باشد. این الگوریتم از یک جستجوی تصادفی استفاده می‌کند که نه تنها تغییراتی که منجر به بهبود تابع هدف می‌شود را می‌پذیرد، بلکه برخی از تغییراتی را که منجر به عدم بهبود تابع هدف می‌شود را نیز موجه می‌داند.

در هر قدم از الگوریتم متروپولیس به یک ذره اجازه جا به جایی کوچکی داده می‌شود و تغییرات ایجاد شده در انرژی سیستم محاسبه می‌گردد که این تغییرات با Δf نشان داده می‌شود. در صورتی $\Delta f \leq 0$ که جا به جایی پذیرفته می‌شود و چنانچه $\Delta f > 0$ اتفاق بیفتد، با فرایند به صورت احتمالی برخورد می‌شود. احتمال پذیرش یک حالت بدتر[1] به وسیله رابطه زیر محاسبه می‌شود. در هر دما الگوریتم به تعداد تکرار مشخصی انجام می‌شود و سپس درجه حرارت کاهش می‌یابد. این فرایند تا زمانی که سیستم به یک حالت پایدار برسد ادامه می‌یابد.

$$P = \exp\left(-\frac{\Delta f}{T}\right) > r$$

Δ : میزان تغییرات در تابع هدف

T : دمای فعلی

r : عدد تصادفی که توسط توزیع یکنواخت در فاصله $[0,1]$ به دست می‌آید.

پس از محاسبه مقدار p، یک عدد تصادفی در فاصله $[0,1]$ تولید می‌شود. در صورتی که احتمال p بزرگ‌تر از عدد تصادفی r باشد، حرکت بدتر پذیرفته و در غیر این صورت رد خواهد شد. بنابراین احتمال پذیرش یک حرکت بدتر تابعی از دو مقدار درجه حرارت سیستم و میزان تغییرات تابع هدف می‌باشد. هر چه درجه حرارت سیستم کاهش یابد، احتمال پذیرش یک حرکت بدتر نیز کاسته می‌شود. در یک حالت حدی اگر درجه حرارت صفر باشد، تنها حرکات بهتر پذیرفته خواهند شد. تعدیل دما و الگوهای کاهشی آن در زمانبندی سرد شدن جهت کنترل رفتار الگوریتم شبیه‌سازی تبرید به کار می‌روند. به منظور اجتناب از گیر افتادن در دام بهینه محلی، جواب‌هایی که مقدار تابع هدف را بدتر می‌کنند نیز تحت مکانیزم تصافی

1- Inferior

مشخصی مورد پذیرش واقع می‌شوند. هر چه این مکانیزم به سمت جلو حرکت می‌کند دما به صورت تدریجی کاهش می‌یابد.

نتایج محاسباتی

در این بخش، با استفاده از آزمایشات عددی، عملکرد مدل‌های ریاضی و الگوریتم‌های حل ارزیابی می‌شود. در این راستا، دو آزمایش مختلف طراحی می‌شود که آزمایش اول شامل تعدادی مساله در اندازه کوچک است. مسایل به صورت بهینه توسط نرم‌افزار تخصصی بهینه‌سازی حل می‌شوند. آزمایش دوم شامل مسایلی با اندازه بزرگ‌تر است که عملکرد الگوریتم رقابت استعماری را با الگوریتم شبیه‌سازی تبرید مقایسه می‌کند. برای انجام این آزمایش‌ها از نرم‌افزار بهینه‌سازی CPLEX برای حل مدل‌های ریاضی و از نرم افزار Matlab برای کد کردن الگوریتم‌های فراابتکاری استفاده می‌شود. از کامپیوتری با مشخصات Core2Due و ۲ گیگاهرتز حافظه استفاده می‌شود.

مشخصات مسایل در اندازه کوچک شامل، تعداد کارها ۴، ۶ و ۸، تعداد مراحل مختلف کاری ۳ و ۵ و تعداد ماشین موازی در هر مرحله ۲ در نظر گرفته شده است. لذا ۶ ترکیب از تعداد کارها، تعداد مراحل مختلف کاری و تعداد ماشین موازی در هر مرحله به‌وجود می‌آید و ۲ مساله برای هر ترکیب ایجاد شده که در مجموع ۱۲ مساله تولید می‌شود. زمان پردازش کارها در هر مرحله به طور تصادفی از توزیع یکنواخت در بازه ۱۰ تا ۹۹ تولید می‌شود. برای تولید مسایل در اندازه‌های بزرگ، تعداد کارها برابر با ۲۰، ۵۰ و ۱۰۰ و همچنین تعداد ایستگاه‌ها را ۵ و ۱۰ لحاظ می‌شود. تعداد ماشین‌ها در هر ایستگاه نیز در حالت اول برای همه ایستگاه‌ها ۲ و در حالت دوم از یک توزیع یکنواخت بین ۱ تا ۳ تولید می‌شود. برای هر یک از ۱۲ ترکیب فوق ۵ مثال تولید می‌شود که مجموع آنها ۶۰ عدد خواهد شد.

ارزیابی مدل‌های ریاضی پیشنهادی

در این بخش، عملکرد دو مدل ریاضی ارائه شده ارزیابی و مقایسه می‌شوند. برای ارزیابی عملکرد مدل‌های برنامه‌ریزی ریاضی، پیچیدگی اندازه و پیچیدگی محاسباتی مدل‌ها بررسی می‌شود. برای اطلاعات بیشتر خوانندگان به مقاله‌های استفورد و همکاران[1] (۲۰۰۵)، تسنگ و استفورد[2] (۲۰۰۸) و نادری و سلماسی[3] (۲۰۱۲) ارجاع داد.

به دنبال بررسی مدل از نظر پیچیدگی اندازه[4] آن برای مدل‌سازی یک مسئله است. برای ارزیابی این مدل، آن را از طریق پیچیدگی اندازه در مشخصه‌های اصلی بررسی می‌شود. مشخصه‌های اصلی یک مدل MILP تعداد متغیرهای جفتی (باینری)، تعداد متغیرهای پیوسته و تعداد محدودیت‌های لازم برای فرموله کردن یک مسئله هستند. برای انجام این کار، تعداد متغیرهای باینری، تعداد متغیرهای پیوسته و تعداد محدودیت‌های مورد نیاز برای مدل‌سازی یک مسئله با تعداد n کار، g ایستگاه و m ماشین در هر ایستگاه محاسبه می‌شود. جدول ۱ نتایج را نشان می‌دهد. مدل ۱، تعداد متغیرهای باینری کمتر و همچنین محدودیت‌های کمتری را برای مدل‌سازی یک مسئله نسبت به مدل ۲ نیاز دارد. تعداد متغیرهای پیوسته دو مدل یکسان است.

جدول ۱. تعداد متغیرهای باینری و پیوسته و محدودیت مدل‌ها

تعداد	مدل ۱	مدل ۲
متغیر باینری	$\frac{n(n-1)}{2}g + ngm$	n^2gm
متغیر پیوسته	ng	ng
محدودیت	$2ng + n(n-1)gm + n$	$3ng + gm + ngm + 2n^2g + n$

1 -Stafford et al.

2 -Teseng and Stafford

3- Naderi and Salmasi

4- Size complexity

یکی دیگر از شاخص‌های مهم برای ارزیابی مدل‌های برنامه‌ریزی ریاضی، تخمین زمان محاسباتی مدل برای حل یک مسئله در یک نرم‌افزار تخصصی برنامه‌ریزی ریاضی است. بدیهی است هر مدلی که زمان محاسباتی کمتری نیاز داشته باشد، مدل بهتری محسوب می‌شود. لازم به یادآوری است که این شاخص پیچیدگی محاسباتی مکمل شاخص پیچیدگی اندازه است. باید دقت کرد مدلی که پیچیدگی اندازه کمتری دارد لزوما پیچیدگی محاسباتی کمتری ندارد. زیرا ممکن است با اضافه کردن محدودیت و یا متغیرهای کمکی می‌توان پیچیدگی محاسباتی را کم اما طبیعتا پیچیدگی اندازه بزرگتر می‌شود. همچنین همان‌طور که پیش‌تر مطرح شد وجود M بزرگ نیز تاثیرگذار است.

برای ارزیابی و مقایسه دو مدل پیشنهادی، مدل‌ها در نرم‌افزار سیپلکس[1] کد و با کامپیوتری با 2GB RAM و CPU Core2due حل می‌شوند. یک مجموعه مثال عددی شامل ۶ مثال از مسئله در اندازه‌های مختلف تولید می‌شود تا توسط مدل‌ها حل شوند. در این مثال‌ها تعداد کارها برابر با ۶، ۸ و ۱۰ در نظر می‌گیریم. تعداد ایستگاه‌ها برابر با ۴ و ۶ خواهد بود. لازم به ذکر است که زمان‌های پردازش از یک توزیع یکنواخت بین ۱ تا ۷۰ تولید می‌شوند. برای هر مثال حداکثر ۳۶۰۰ ثانیه به مدل‌ها زمان داده می‌شود. جدول ۲ نتایج حاصل از حل این مثال‌ها در نرم‌افزار توسط هر دو مدل را نشان می‌دهد. اطلاعات این جدول شامل موارد زیر است: زمان مورد نیاز برای مثال‌هایی که در کمتر از ۳۶۰۰ ثانیه به صورت بهینه حل شده‌اند و شکاف بهینگی (فاصله بین حد پایین و بالا) برای مثال‌هایی که در ۳۶۰۰ ثانیه به صورت بهینه حل نشده‌اند.

مدل اول همه ۱۲ مثال را به صورت بهینه حل نمود در صورتی که مدل دوم ۸ مثال کوچک‌تر را به صورت بهینه حل می‌کند. در اصل، به نظر می‌رسد مدل اول مسائل با اندازه بزرگی ۱۰ کار را می‌تواند حل کند و در حالی که مدل دوم حداکثر مثال‌هایی با اندازه ۸ کار را می‌تواند به صورت بهینه حل کند. متوسط شکاف بهینگی مدل دوم برای اندازه ۱۰ کار برابر است

1- CPLEX

۴۶٪. از آنجایی که مدل اول همه مسایل به صورا بهینه حل می‌کند، شکاف بهینگی آن برای همه مثال‌ها صفر است.

جدول ۲. نتایج محاسبه پیچیدگی محاسباتی مدل‌ها

مثال			مدل ۱			مدل ۲		
n	M	تابع هدف	شکاف بهینگی	زمان حل (ثانیه)	تابع هدف	شکاف بهینگی	زمان حل (ثانیه)
۶	۳	۴۲۷	۰	<۱	۴۲۷	۰	<۱
۶	۳	۵۱۲	۰	<۱	۵۱۲	۰	<۱
۶	۵	۶۸۳	۰	<۱	۶۸۳	۰	<۱
۶	۵	۷۳۹	۰	<۱	۷۳۹	۰	<۱
۸	۳	۶۴۷	۰	<۱	۶۴۷	۰	۲۵
۸	۳	۸۲۱	۰	<۱	۸۲۱	۰	۱۲
۸	۵	۱۰۷۷	۰	۳	۱۰۷۷	۰	۹۸
۸	۵	۹۸۰	۰	۵	۹۸۰	۰	۵۴
۱۰	۳	۸۵۱	۰	۱۵	۸۵۵	٪۴۰	۳۶۰۰
۱۰	۳	۱۰۹۳	۰	۳۱	۱۰۹۴	٪۳۵	۳۶۰۰
۱۰	۵	۱۲۶۷	۰	۲۳۶	۱۳۹۲	٪۵۲	۳۶۰۰
۱۰	۵	۱۳۱۰	۰	۳۷۱	۱۳۴۲	٪۵۷	۳۶۰۰

ارزیابی الگوریتم‌های پیشنهادی

ابتدا مسایلی که با اندازه کوچک تولید شده‌اند را توسط الگوریتم رقابت استعماری (ICA) و شبیه‌سازی تبرید (SA) حل می‌شود. در ادامه مسایل با اندازه بزرگ توسط هر دو الگوریتم و با تحلیل نتایج عملکرد الگوریتم‌ها ارزیابی و مقایسه می‌شود.

قبل از انجام آزمایش‌های مربوط به الگوریتم‌ها، پارامتر هر یک تنظیم می‌شود. الگوریتم رقابت استعماری یک پارامتر $nEmp$ دارد و همچنین الگوریتم شبیه‌سازی تبرید نیز داری پارامتر ترکیبی (T_0, α) است. برای هر یک از این پارامترها ۴ سطح در نظر گرفته می‌شود. با حل مثال‌های تولید شده، در نهایت $nEmp$ برابر ۴۰ و (T_0, α) برابر با (۰,۹۵ و ۱۰۰) بهترین عملکرد را داشتند.

ابتدا عملکرد این الگوریتم‌ها را در مقابل جواب بهینه به‌دست آمده توسط مدل‌های ریاضی مقایسه می‌کنیم. نتایج در جدول ۳ نشان داده شده است. همانطور که نتایج نشان می‌دهد الگوریتم رقابت استعماری عملکرد بهتری در مقایسه با الگوریتم شبیه‌سازی تبرید دارد. الگوریتم رقایت استعماری ۹ تا از ۱۲ مثال به صورت بهینه حل می‌کند در صورتی که الگوریتم شبی‌سازی تبرید ۷ مثال را به صورت بهینه حل می‌کند.

جدول ۳. نتایج الگوریتم‌ها در اندازه‌های کوچک

n	m	مدل	ICA	SA
۶	۳	۴۲۷	۴۲۷	۴۲۷
۶	۳	۵۱۲	۵۱۲	۵۱۲
۶	۵	۶۸۳	۶۸۳	۶۸۳
۶	۵	۷۳۹	۷۳۹	۷۳۹
۸	۳	۶۴۷	۶۴۸	۶۴۸
۸	۳	۸۲۱	۸۲۱	۸۲۱

۸	۵	۱۰۷۷	۱۰۷۷	۱۰۸۰
۸	۵	۹۸۰	۹۸۲	۹۸۲
۱۰	۳	۸۵۱	۸۵۱	۸۵۹
۱۰	۳	۱۰۹۳	۱۱۰۵	۱۱۳۲
۱۰	۵	۱۲۶۷	۱۲۶۷	۱۲۹۵
۱۰	۵	۱۳۱۰	۱۳۱۵	۱۳۱۵

در ادامه، با استفاده از مثال‌های اندازه بزرگ، عملکرد الگوریتم پیشنهادی رقابت استعماری با الگوریتم شبیه‌سازی تبرید مقایسه می‌شود. تمامی مسایل با الگوریتم رقابت استعماری و شبیه-سازی تبرید حل شده است و درصد انحراف نسبی[1] یا RPD آنها به‌دست می‌آید.

$$RPD = 100(Sol - Min)/Min$$

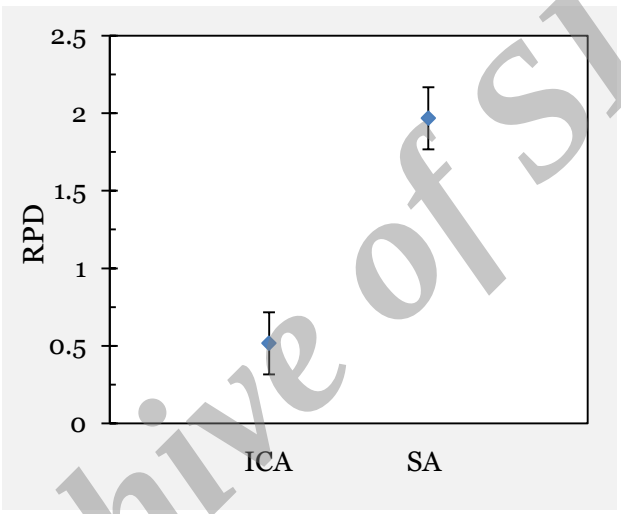
جایی که Sol مقدار تابع هدف به‌دست آمده توسط الگوریتم در یک مثال و Min کمترین مقدار میکسپن بدست آمده برای آن مثال است. نتایج در جدول ۴ نشان داده شده است. الگوریتم رقابت استعماری با متوسط RPD برابر با ۰/۵۲٪ عملکردی بهتری نسبت به الگوریتم شبیه‌سازی تبرید با RPD برابر با ۱/۹۷٪ دارد.

جدول ۴. نتایج RPD الگوریتم‌ها در اندازه‌های بزرگ

n	m	ICA	SA
۲۰	۵	۰/۶	۱/۳
۲۰	۱۰	۰/۹	۱/۸
۵۰	۵	۰/۵	۱/۷
۵۰	۱۰	۰/۴	۱/۹
۱۰۰	۵	۰/۴	۲/۴
۱۰۰	۱۰	۰/۳	۲/۷
	متوسط	۰/۵۲	۱/۹۷

1- Relative Precentage Deviation

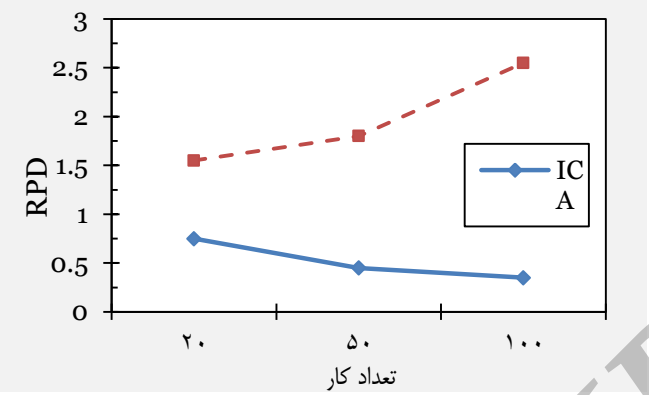
برای بررسی اختلاف بین میانگین الگوریتم‌ها، تست آنالیز واریانس یک‌طرفه بر روی مقادیر درصد انحراف نسبی اجرا شده است. با توجه به صفر بودن مقدار P عدم برابری میانگین الگوریتم‌ها را می‌توان نتیجه گرفت. برای بررسی عملکرد الگوریتم‌ها از میانگین و بازه حداقل تفاوت معنادار (LSD)[1] بر روی مقادیر درصد انحراف نسبی به‌دست آمده در حل مسایل تصادفی، استفاده شده است. همان‌طور که در شکل ۲ مشخص است الگوریتم رقابت استعماری عملکرد بهتری نسبت به شبیه‌سازی تبرید برای حل مساله مورد بررسی دارد.



شکل ۲. نمایش میانگین و بازه LSD

برای تحلیل تاثیر مقادیر تعداد کار بر روی عملکرد الگوریتم‌ها به تحلیل نتایج پرداخته شده است. شکل ۳ میانگین انحراف نسبی به‌دست آمده از الگوریتم‌ها را در مقادیر مختلف تعداد کار نشان می‌دهد. الگوریتم رقابت استعماری به ازای افزایش تعداد کارها عملکرد بهتر و پایدارتری از خود ارائه می‌دهد.

1- Least Significant Difference



شکل ۳. نمایش عملکرد الگوریتم‌ها به ازای تعداد کارهای مختلف

نتیجه‌گیری

در این مقاله مسئله زمانبندی جریان کاری ترکیبی با ماشین‌های موازی در هر یک از ایستگاه‌های کاری با وجود کارهای بدون انتظار مطالعه شد. کارهای بدون انتظار کارهایی محسوب می‌شوند که باید بین پردازش عملیات‌های آن کار نباید هیچ فاصله زمانی وجود داشته باشد. در ادبیات همه مقالات، همه کارها را بدون تاخیر در نظر می‌گیرند در صورتی در ممکن است در کارگاه تنها بعضی از کارها چنین ویژگی داشته باشند. در نتیجه در این مسئله کارها به دو گروه تقسیم شده‌اند: کارهایی که باید به صورت بدون انتظار زمانبندی شوند و کارهای معمولی.

در این مقاله، ، دو مدل ریاضی قالب برنامه‌ریزی عددصحیح خطی مختلط توسعه داده شد. با استفاده از نرم‌افزارهای تجاری تخصصی تحقیق در عملیات مدل ریاضی حل و نتایج عملکردی آن تحلیل و مقایسه شد. دو شاخص‌های ارزیابی عملکرد مدل‌های ریاضی، پیچیدگی اندازه و پیچیدگی محاسباتی استفاده شد. سپس برای حل مسئله در اندازه‌های واقعی، دو الگوریتم فراابتکاری شامل الگوریتم رقابت استعماری و شبیه‌سازی تبرید توسعه داده شد. یک مجموعه مثال آزمایشگاهی تولید و عملکرد الگوریتم‌ها با یکدیگر مقایسه می‌شود. الگوریتم رقابت استعماری در مقایسه با الگوریتم دیگر عملکرد بهتری ارایه کرد.

تقدیر و تشکر

از معاونت پژوهشی دانشگاه خوارزمی بابت حمایت از این پژوهش تقدیر و تشکر می‌شود.

مراجع

Aldowaisan, T., Allahverdi, A., (2004). *New heuristics for m-machine no-wait flowshop to minimize total completion time*. Omega, 32, 345–352.

Atashpaz-Gargari, E., Lucas, C., (2007) *Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition.* IEEE Congress, 4661–4670.

Attar, S.F., Mohammadi, R., Tavakkoli-Moghaddam, M., (2011). 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, 27-32.

Chang, J.L., Gong, D.W., MA, X.P., (2007). A heuristic genetic algorithm for no-wait flowshop scheduling problem, Journal of China University of Mining and Technology, 17, 582–586.

Chen, C., Neppalli, R., (1996). *Genetic algorithms applied to the continuous flow shop problem.* Computers and Industrial Engineering, 30, 919–929.

Fink, A., Voß, S., (2003). Solving the continuous flow-shop scheduling problem by metaheuristics. European Journal of Operational Research, 151, 400–414.

Hall, N.C., Sriskandarajah, C.A., (1996). *Survey of machine scheduling problems with blocking and no-wait in process,* Operations Research, 44, 510–525.

Giaro, K., (2001). NP-hardness of compact scheduling in simplified open and flow shops, European Journal of Operational Research, 130, 90–98.

Goyal, S.K., Sriskandarajah, C., (1998). *No-wait shop scheduling: Computational complexity and approximate algorithms,* Operations Research, 25, 220–244.

Laha, D., Sapkal, S.U., (2014). *An improved heuristic to minimize total flow time for scheduling in them-machine no-wait flow shop*, Computers and Industrial Engineering, 67, 36-43.

Naderi, B., Salmasi, N., (2012). Permutation flowshops in group scheduling with sequence-dependent setup times, European Journal of Industrial Engineering, 6(2), 177-199.

Naderi, B., Khalili, M., Khamseh, A.A., (2014*). Mathematical models and a hunting search algorithm for the no-wait flowshop scheduling with parallel machines,* International Journal of Production Research, 52, 2667-2681.

Nagano, M.S., da Silva, A.A., Lorena, L.A.N., (2014). *An evolutionary clustering search for the no-wait flow shop problem with sequence dependent setup times,* Expert Systems with Applications, 41, 3628-3633.

Schuster, C.J., Framinan, J.M., *Approximate procedure for no-wait job shop scheduling.* Operations Research Letters, 31, 308–318.

Rajendran, C., (1994). A no-wait flowshop scheduling heuristic to minimize makespan, Journal of the Operational Research Society, 45, 472–478.

Röck, H., (1984). *Some new results in flow shop scheduling, Mathematical Methods of Operations Research,* 28, 1–16.

Ruiz, R., Allahverdi, A., (2007). *Some effective heuristics for no-wait flowshops with setup times to minimize total completion time*, Annals of Operations Research, 156, 143–171.

Pan, Q.K., Wang, L., Zhao, B.H., (2008). An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion, International Journal of Advanced Manufacturing Technology, 38, 778-786.

Stafford, E.F., Tseng, F.T., Gupta, J.N.D., (2005). Comparative evaluation of MILP flowshop models, Journal of Operational Research Society, 56, 88–101.

Tseng, F.T., Stafford, E.F., (2008). New MILP models for the permutation flowshop problem, Journal of the Operational Research Society, 59, 1373–1386.