زمان‌بندی سیستم‌های تولید کارگاهی انعطاف‌پذیر با استفاده از الگوریتم جستجوی فاخته بهبودیافته با خوشه‌بندی مارکوف و پرواز لوی

زینب صادقی چوینلی[۱]، دانشجو، سیدمحمدحسین معطر[۲]، استادیار
۱-دانشکده فنی و مهندسی - دانشگاه آزاد اسلامی واحد مشهد- مشهد- ایران - zeinabsadeghi@mshdiau.ac.ir
۲- دانشکده فنی و مهندسی- دانشگاه آزاد اسلامی واحد مشهد- مشهد - ایران - moattar@mshdiau.ac.ir

چکیده: با توجه به پیچیدگی بالای مسائل زمان‌بندی، روش‌های کلاسیک جواب‌گوی حل این مسئله نیستند، بنابراین امروزه از الگوریتم‌های فرااکتشافی در حل آن استفاده می‌شود. در این مقاله الگوریتم بهینه‌سازي فاخته به‌عنوان یکي از جدیدترین و قوي‌ترین روش‌هاي بهینه‌سازي تکاملي برای حل مسئله زمان‌بندی کارکارگاهی انعطاف‌پذیر استفاده شده است. در الگوریتم پیشنهادی برای بهبود پاسخ‌ها، ترتیب ورود جمعیت اولیه بر اساس الگوریتم NEH-D، که مبتنی بر کاهش زمان اجرای هر یک از کارها است، تعیین شده است. سپس ماشین‌های فعال توسط خوشه‌بندی مارکوف گروه‌بندی می‌گردند، تا در هر مرحله از عملیات، انتخاب ماشین از بین ماشین‌های فعال صورت گیرد. بنابراین تعداد جواب‌های انتخابی برای الگوریتم جستجوی فاخته محدود می‌گردد، تا سرعت اجرای الگوریتم فاخته افزایش یابد. درنهایت نیز از الگوریتم جستجوی فاخته برای تخصیص ماشین‌ها به کارها و از پرواز لوی برای بهبود در الگوریتم فاخته جهت جستجوی سراسری در کنار جستجوی محلی استفاده شده است. الگوریتم پیشنهادی بر روی مجموعه داده استاندارد Kacem، Brandimarte و داده‌های مقالات مرتبط ارزیابی شده است. نتایج تجربی نشان می‌دهد، که الگوریتم پیشنهادی سرعت بالاتری در رسیدن به جواب نهایی و همچنین همگرایی بالایی در جواب‌ها دارد.

واژه‌های کلیدی: زمان‌بندی کار کارگاهی انعطاف‌پذیر، الگوریتم جستجوی فاخته، الگوریتم NEH-D، جستجوی همسایگی، خوشه‌بندی مارکوف، پرواز لوی.

Flexible job shop scheduling using improved cuckoo search algorithm by Markov clustering and Levy flight

Z. Sadeghi, Student[1], S. M. H. Moattar, Assistant professor[2]
1-Computer Engineering Department, Mashhad branch, Islamic Azad University, Mashhad, Iran, Email: zeinabsadeghi@mshdiau.ac.ir
2- Computer Engineering Department, Mashhad branch, Islamic Azad University, Mashhad, Iran, Email: moattar@mshdiau.ac.ir

Abstract: Considering the high complexity of scheduling problems, classic approaches fail to find the solution efficiently. Therefore, meta-heuristic algorithms are used for this purpose. In this paper, Cuckoo optimization algorithm (COA) is used as one of the novel and most effective evolutionary optimization algorithms for flexible job shop scheduling. In the proposed approach, for better solutions, the initial population is determined using NEH-D algorithm, which considers the completion time minimization of each job. Then active machines are grouped using Markov clustering, so that the assigned machine is chosen from the active ones, hoping that the possible solutions of COA are bounded and the execution speed of the algorithm is increased. Finally, COA is applied for job-machine assignment and Levy flight is used to improve the global search of the algorithm. The proposed approach is evaluated on standard datasets such as Kacem, Brandimarte and other related data. The experimental results show that the proposed algorithm is capable of finding the final solution with lower computational complexity and has higher convergence rate.

Keywords: Flexible job shop scheduling, cuckoo search algorithm, NEH-D algorithm, neighborhood search, Markov clustering, Levy flight.

تاریخ ارسال مقاله: ۹۴/۹/۲
تاریخ اصلاح مقاله: ۹۴/۱۱/۱۸ و ۹۵/۱/۲۵
تاریخ پذیرش مقاله: ۹۵/۳/۲۶
نام و نام خانوادگی نویسنده مسئول: سیدمحمدحسین معطر
نشانی نویسنده مسئول: مشهد، دانشگاه آزاداسلامی واحد مشهد، دانشکده فنی و مهندسی

۱- مقدمه

زمان‌بندی، تخصیص منابع در طول زمان برای اجرای مجموعه‌ای از وظایف است. این تعریف دو مفهوم مختلف را دربر دارد. اولاً زمان‌بندی نوعی تصمیم‌گیری است و فرایندی است که در جریان آن برنامه زمانی تعیین می‌شود. ثانیاً زمان‌بندی مبحثی نظری است که مجموعه‌ای از اصول، مدل‌ها، روش‌ها و نتایج منطقی را در بر می‌گیرد، که برای ما بینشی عمیق در مورد عمل زمان‌بندی فراهم می‌آورد.

سه هدف در زمان‌بندی عمده‌تر به‌نظر می‌رسند: بهره‌برداری کارا از منابع، پاسخ‌گویی سریع به تقاضا و انطباق دقیق موعدهای تحویل تعیین شده. غالباً می‌توان از یك معیار مهم هزینه‌ای مربوط به سنجش عملکرد سیستم (مانند زمان بیکاری ماشین، زمان انتظار برای انجام کار یا تاخیر کار) به‌عنوان جانشینی برای هزینه کل سیستم استفاده کرد [۱]. زمان‌بندی سیستم تولید کارگاهی انعطاف‌پذیر[۱] شامل زمان‌بندی n کار روی m ماشین است. هر کار دارای تعدادی عملیات است و برای هر عملیات، امکان استفاده از یک مجموعه ماشین وجود دارد. زمان‌بندی سیستم‌های تولید کارگاهی انعطاف‌پذیر به دلیل جایگاه ویژه آن در مراکز تولیدی مورد توجه زیاد مدیران واحدهای تولیدی است. شکل ساده مسئله زمان‌بندی تولید کارگاهی انعطاف‌پذیر مسئله زمان‌بندی تولید کارگاهی کلاسیک است، که به‌صورت زمان‌بندی n کار $J_1, J_2, ..., J_n$ روی مجموعه M از ماشین‌ها شامل m ماشین $M_1,M_2,...,M_m$ تعریف می‌شود. هر کار دارای O_j عملیات است که بایستی به ترتیب انجام شوند. هدف زمان‌بندی در این مسئله، تعیین توالی عملیات برای هر ماشین است، به‌نحوی‌که یک تابع هدف از قبل مشخص‌شده مثل دوره ساخت، بهینه گردد [۲].

ادبیات مطرح‌شده در زمینه سیستم‌های تولید کارگاهی انعطاف‌پذیر بسیار گسترده است. بروکر و همکاران [۳] اولین افرادی بودند که این مسئله را مورد بررسی قرار دادند. آن‌ها یک الگوریتم چندوجهی براي حل مسئله کارگاهی انعطاف‌پذیر با دو کار ارائه کردند. روش‌های فراابتکاری توسعه داده‌شده در سال‌های اخیر، روش‌های جایگزین بسیار مناسب و جذاب برای حل مسائل زمان‌بندی جریان کارگاهی به حساب می‌آیند.

در سال ۲۰۱۰، زینگ [۴] الگوریتم بهینه‌سازی مورچگان مبتنی بر دانش را برای مسئله زمان‌بندی فروشگاه کار انعطاف‌پذیر پیشنهاد کرده است. الگوریتم KBACO[۲] ادغام مدل بهینه‌سازی مورچگان و مدل مبتنی بر دانش است. در الگوریتم آن‌ها مدل دانش، برخی از دانش بهینه‌سازی مورچگان را یاد می‌گیرد و سپس دانش موجود را برای هدایت جستجوی اکتشافی فعلی اعمال می‌کند. تکنگ در سال ۲۰۱۲ [۵] الگوریتم جهش قورباغه برای حل مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر پیشنهاد کرده است. مدل پیشنهادی در [۵] ترکیبی از الگوریتم جهش قورباغه و مفهوم منطق فازی است.

در سال ۲۰۱۲، ژانگ [۶] الگوریتم رقابت استعماری را برای مسائل زمان‌بندی کارکارگاهی پیشنهاد کرده است و عملیات جستجوی محلی را برای بهبود کیفیت راه‌حل‌ها اجرا کرده است. در سال ۲۰۱۲، جولای [۷] یک الگوریتم رقابت استعماری ترکیبی برای مسئله زمان‌بندی فروشگاه جریان انعطاف‌پذیر با در نظر گرفتن زمان راه‌اندازی، به‌منظور به حداقل رساندن حداکثر زمان اتمام برنامه ارائه کرده است.

برنوال در سال ۲۰۱۳ [۸] روشی مبتنی بر جستجوی فاخته برای بهینه‌سازی زمان‌بندی سیستم‌های تولید انعطاف‌پذیر با به حداقل رساندن جریمه تأخیر تولید و به حداکثر رساندن زمان استفاده از ماشین توسعه داده است. بابوکارتیک در سال ۲۰۱۲ [۹] الگوریتم ترکیبی از بهینه‌سازی کلونی مورچه (ACO) [۳] و جستجوی فاخته برای مسئله زمان‌بندی کار پیشنهاد کرده است. جستجوی فاخته می‌تواند جستجوی محلی مؤثرتر انجام دهد و زمان کل را حداقل کند. در ۲۰۱۴، العبیدی [۱۰] الگوریتم جستجوی ممنوعه بر اساس جستجوی فاخته را پیشنهاد کرده است. الگوریتم جستجوی ممنوعه بستگی به ذخیره‌سازی بهترین همسایه فعلی در بهترین لیست راه‌حل‌ها دارد. از دیگر الگوریتم‌هایی که به‌منظور حل مسئله زمان‌بندی فازی پیشنهاد شده است، می‌توان به [۱۱] اشاره نمود که از الگوریتم بهینه‌سازی مبتنی بر جغرافیای زیستی[۴] برای این منظور استفاده کرده است. در [۱۱] از یک الگوریتم جستجوی محلی مبتنی بر درج به‌منظور بهبود پاسخ‌ها استفاده شده است.

در [۱۲] با هدف بهبود معایب الگوریتم بهینه‌سازی ازحام ذرات[۵] (PSO) و جلوگیری از افتادن الگوریتم در دام بهینه محلی از دو ایده استفاده از عملگر جهش و همچنین استفاده از نقشه آشوب لاجستیک[۶] استفاده شده است و نتایج قابل‌قبولی در مقایسه با سایر الگوریتم‌های زمان‌بندی حاصل شده است. در [۱۳] از یک توسعه بر الگوریتم ژنتیک توزیع‌شده رقابتی چندفضایی[۷] برای تسریع و بهبود جستجو استفاده شده است و برای این منظور الگوریتم به صورت سلسله‌مراتبی اجرا می‌شود. در این مقاله ادعا شده است و نتایج نیز به‌خوبی تبیین می‌کنند، که استفاده از ایده سلسله مراتبی باعث بهبود پاسخ‌ها و افزایش سرعت همگرایی می‌شود. البته روش‌های پیشنهادی در [۱۲] و [۱۳] علی‌رغم مؤثر بودن تنها برای حل مسئله تک‌هدفه پیشنهاد شده‌اند.

در [۱۴] الگوریتم بهبودیافته جستجوی فاخته با مقداردهی اولیه توسط الگوریتم NEH برای مسئله زمان‌بندی جریان کار کارگاهی ترکیبی پیشنهاد شده است و استفاده از روش NEH برای مقداردهی اولیه بسیار مؤثر گزارش شده است. در یک ایده مشابه در [۱۵] از روشی تحت عنوان زمان راه‌اندازی وابسته به توالی[۸] برای مقداردهی اولیه در دو نسخه چندهدفه از الگوریتم ژنتیک استفاده کرده است.

البته استفاده از رویکردهای بهینه‌سازی مکاشفه‌ای در حوزه برنامه‌ریزی و مدیریت پروژه تنها با زمان‌بندی محدود نمی‌شود. برای مثال در [۱۶] از الگوریتم PSO به‌منظور برنامه‌ریزی توسعه شبکه توزیع استفاده شده است. در همین مقاله از شبیه‌سازی مونت کارلو برای رفع مشکل عدم‌قطعیت در برنامه‌ریزی استفاده شده است. همچنین در

[۱۷] از الگوریتم غذایابی باکتری[۹] جهت‌داده‌شده با PSO، در همین مسئله توزیع و برای تعیین مقدار و مکان بهینه خازن‌ها استفاده شده است. دلیل عمده استفاده از رویکرد PSO در الگوریتم مذکور اجتناب از حرکت‌های تصادفی و بهبود زمان پاسخ بوده است.

با توجه به اینکه در برخی تحقیقات گذشته الگوریتم جستجوی فاخته به شکل مؤثری برای مسئله زمان‌بندی استفاده شده است، در این مقاله روشی مبتنی بر بهبود الگوریتم فاخته به‌منظور حل مسئله زمان‌بندی کار کارگاهی پیشنهاد شده است. در روش پیشنهادی استفاده از الگوریتم NEH-D جهت بهبود مقداردهی اولیه و تسریع همگرایی پیشنهاد شده است. همچنین از خوشه‌بندی مارکوف بر روی ماشین‌ها جهت کاهش فضای پاسخ و پرواز لوی جهت افزایش تنوع پاسخ‌ها استفاده شده است. ادامه این مقاله به‌صورت زیر سازمان‌دهی شده است. در بخش ۲ ابتدا مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر معرفی می‌گردد. در بخش ۳ ابزارهای پایه و اولیه مورداستفاده در روش پیشنهادی ذکر شده است. بخش ۴ روش پیشنهادی را معرفی کرده است. ارزیابی روش پیشنهادی در بخش ۵ و جمع‌بندی و نتیجه‌گیری در بخش ۶ ارائه شده است.

۲- طرح مسئله

مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر شامل تخصیص مجموعه‌ای از کارها به مجموعه‌ای از ماشین‌ها با شرایط خاص است و در رده مسائل NP-hard قرار می‌گیرد [۲].

مسئله F-JSSP، حالت توسعه‌یافته زمان‌بندی کار کارگاهی (JSSP)[۱۰] است، که در آن هر ماشین توانایی ارائه بیش از یک عملیات را دارد. بر طبق [۵]، انعطاف‌پذیری در کار کارگاهی به انعطاف‌پذیری ماشین اشاره دارد که ممکن است جزئی (مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر جزئی (PF-JSSP)[۱۱]) یا کلی (مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر کلی (TF-JSSP)[۱۲]) باشد. PF-JSSP یک حالت خاص از F-JSSP است، که در آن امکان استفاده از برخی از ماشین‌ها برای انجام عملیات وجود دارد. درPF-JSSP تعداد مشخصی از ماشین‌های چندمنظوره در سراسر کارگاه توزیع شده است، که انعطاف‌پذیری و چندکاره‌بودن آن‌ها یکسان نیست. این ویژگی این امکان را بوجود می‌آورد که یک قطعه خاص حداقل توسط یک ماشین از میان مجموعه ماشین‌های موجود پردازش شود. در PF-JSSP، m ماشین برای پردازش n کار وجود دارد، که هر کار j شامل n_j عملیات است که باید توسط ماشین‌های موجود انجام شود. هر عملیات $O_{j,i}$ می‌تواند روی تعدادی از ماشین‌های مشابه یا غیرمشابه پردازش شود و زمان پردازش می‌تواند بر اساس مشخصات ماشین متفاوت باشد.

اصلی‌ترین هدف این تحقیق، زمان‌بندی کارها و اختصاص آن‌ها به ماشین‌ها است به‌طوری‌که زمان اتمام کل[۱۳] یعنی مجموع زمان لازم برای انجام همه کارها در آن‌ها به حداقل برسد. همگرایی به جواب بهینه به‌طوری‌که حرکت کلی به‌سمت راه‌حل بهینه باشد و رسیدن به حداقل بار کاری[۱۴] و حداکثر بار کاری از دیگر اهداف این تحقیق می‌باشد.

مفروضات زیر برای مسئله در نظر گرفته شده است:

۱. کارها مستقل هستند و هیچ اولویتی در تخصیص کارها وجود ندارد.
۲. ماشین‌ها مستقل هستند.
۳. تمام ماشین‌ها در زمان صفر در دسترس هستند.
۴. زمان پردازش قطعی و مطلق است.
۵. از زمان راه‌اندازی ماشین‌ها صرف‌نظر شده است.
۶. از زمان انتقال بین عملیات صرف‌نظر شده است.
۷. هر ماشین ظرفیت پردازش یک عملیات در هر زمان را دارد.
۸. هر عملیات حداکثر در یک ماشین در یک زمان اجرا می‌شود.
۹. همپوشانی عملیات مجاز نیست.
۱۰. هیچ شکافی در زمان پردازش وجود ندارد.
۱۱. تمام ماشین‌ها برای تمام مدت زمان‌بندی در دسترس هستند.

۳- ابزارها و روش‌ها

در این بخش روش‌های پایه مورداستفاده در الگوریتم پیشنهادی به تفصیل بیان شده است.

۳-۱- الگوریتم NEH-D

الگوریتم NEH-D توسعه‌یافته الگوریتم NEH [۱۵] می‌باشد که براساس انحراف معیار بهبود یافته است. NEH اکتشافی شامل دو مرحله است: در مرحله اول، کارها برحسب زمان پردازش خود به صورت نزولی مرتب می‌شوند و در مرحله دوم، دنباله کار، با ارزیابی زمان‌بندی حاصل از برنامه حاصل فاز اول، به ترتیب ساخته می‌شود. در پیاده‌سازی، الگوریتم مرتب‌سازی پشته برای فاز اول انتخاب شده است. برای درج کار در فاز دوم، کار همیشه به موقعیت اول با زمان کل حداقل، اضافه می‌گردد.

$$AVG_j = \frac{1}{m}\sum_{i=1}^{m} P_{ij} \qquad (۱)$$

$AVGj$ میانگین زمان پردازش کار j، m تعداد ماشین‌ها و P_{ij} نشان‌دهنده زمان پردازش کار j بر روی ماشین i است.

درحالی‌که فاز دوم با NEH یکسان است، در مرحله اول، AVG_j با استفاده از معادله (۱) [۱۸] تعریف می‌گردد. انحراف استاندارد از زمان پردازش به‌شرح رابطه (۲) است [۱۸].

$$STD_j = \left[\frac{1}{m-1}\sum_{i=1}^{m}\left(P_{ij} - AVG_j\right)^2\right]^{\frac{1}{2}} \qquad (۲)$$

برای درج کار NEH در فاز دوم، ممکن است چندین حالت وجود داشته باشد، به‌عنوان مثال ممکن است چندین توالی جزئی با همان زمان کل جزئی وجود داشته باشد. اما عملکرد بهتر را می‌توان با

استفاده از استراتژی مناسب انتظار داشت. با این فرضیه، استراتژی زیر ارائه شده است:

در مرحله اول، تعریف‌های زیر را داریم:

$\pi(x)$ اشاره به کار در موقعیت x

$P_{i,\pi(x)}$ اشاره به زمان پردازش $\pi(x)$ در ماشین i

$C_{i,\pi(x)}$ اشاره به زمان اتمام کار $\pi(x)$ در ماشین i

$S_{i,\pi(x)}$ آخرین زمان ممکن شروع کار $\pi(x)$ در ماشین i

دو معادله (۳) و (۴) را می‌توان برای کار $\pi(x)$ به‌شرح زیر محاسبه کرد[۱۸]:

$$\mathrm{E}_{\pi(x)} = \frac{1}{m}\sum_{i=1}^{m}\frac{P_{i,\pi(x)}}{S_{i,\pi(x+1)} - C_{i,\pi(x-1)}}, \quad x = 1,\ldots,n \tag{۳}$$

$$D_{\pi(x)} = \sum_{I=1}^{M}\left(\frac{P_{i,\pi(x)}}{S_{i,\pi(x+1)} - C_{i,\pi(x-1)}} - E_{\pi(x)}\right)^2 \quad x = 1,\ldots,n \tag{۴}$$

سپس هنگام قراردادن یک کار در دنباله جزئی، مکانی که دارای زمان کل حداقل باشد ابتدا انتخاب می‌گردد و اگر چندین گزینه وجود داشته باشد، x با حداقل $D_{\pi(X)}$ انتخاب می‌شود.

اگر NEH با ترکیبی از قانون اولویت AVG در فاز اول و استراتژی جدید در فاز دوم اصلاح گردد، می‌توان انتظار عملکرد بهتری داشت. این الگوریتم بهبودیافته، NEH–D نامیده می‌شود. الگوریتم NEH-D به‌معنای NEH بهبودیافته با استفاده از انحراف است، که به شرح زیر ارائه شده است.

۱. محاسبه متوسط زمان پردازش AVG_j و انحراف استاندارد بار پردازش STD_j برای هر کار j و مرتب کردن کارها به ترتیب کاهش $AVG_j + STD_j$
۲. اگر تنها یک کار وجود دارد، به‌عنوان دنباله است.
۳. برای کار kام در میان K حالت ممکن با حداقل زمان کل جزئی، در صورت وجود روابط ذکرشده، x با حداقل $D_{\pi(X)}$ انتخاب می‌شود و به دنباله اضافه می‌گردد.

۳-۲- خوشه‌بندی مارکوف

خوشه‌بندي فرآيند دسته‌بندي مجموعه‌اي از اشياء به خوشه‌هايي است که اعضای دروني هر خوشه بيش‌ترين شباهت را به يکديگر و کم‌ترين شباهت را نسبت به اعضای ساير‌خوشه‌ها داشته باشند. در حالت کلی ممکن است روش‌های متفاوتی برای مشخص کردن خوشه‌بندی دو گره همسایه وجود داشته باشد، اما اکثراً این کار را به‌صورت محلی انجام می‌دهند. خوشه‌بندی مارکوف یک راه‌حل ساده و عمومی برای شناسایی و تفکیک خوشه‌ها از یکدیگر است. این روش براساس عملگرهای تفکیک است، که بر روی گراف مکرراً اعمال می‌شود و منجر به مشخص شدن خوشه‌ها می‌شود. این عمل باعث می‌شود که ارزش یال‌های داخل خوشه‌ها افزایش و ارزش یال‌های بین خوشه‌ها کاهش یابد و درنتیجه خوشه‌ها مشخص می‌گردد. این عملگرها ساده‌تر، سریع‌تر و طبیعی‌تر خوشه‌بندی می‌کنند و در بسیاری از برنامه‌ها مفیدتر هستند[۱۹].

تفکیک بر اساس شباهت همسایگی بین گره i با سایر گره‌ها است. در حقیقت $P_{visit}^{k}(i)$ مقدار (وزن) یال‌هایی است که از i به‌سمت همسایگانش است. ازاین‌رو همسایگی i با $P_{visit}^{1}(i)$ نشان داده‌می‌شود. در اکثر موارد برای تخمین زدن مقدار $P_{visit}^{k}(v)$ و $P_{visit}^{k}(u)$ (شباهت دو گره u و v) از مقدار کوچک k (مثلا ۳=k) استفاده می‌شود. هرچه اختلاف شباهت بین u و v کم‌تر باشد، سبب افزایش وزن ارتباط آن می‌گردد.

۳-۳- الگوریتم جستجوی فاخته[۱۶]

الگوریتم فاخته با الهام از روش زندگی پرنده‌ای بنام فاخته است، که در سال ۲۰۰۹ توسط شین او یانگ[۱۷] و دب ساوش[۱۸]، توسعه یافته است [۲۰]. همانند سایر الگوریتم‌های تکاملی، الگوریتم بهینه‌سازی فاخته هم با یک جمعیت اولیه کار خود را شروع می‌کند، جمعیتی متشکل از فاخته‌ها. این جمعیت از فاخته‌ها تعدادی تخم دارند که آن‌ها را در لانه تعدادی پرنده میزبان می‌گذارند. تعدادی از این تخم‌ها که شباهت بیش‌تری به تخم‌های پرنده میزبان دارند، شانس بیش‌تری برای رشد و تبدیل‌شدن به فاخته بالغ خواهند داشت. سایر تخم‌ها توسط پرنده میزبان شناسایی شده و از بین می‌روند. میزان تخم‌های رشدکرده مناسب‌بودن لانه‌های آن منطقه را نشان می‌دهند. هرچه تخم‌های بیش‌تری در یک ناحیه قادر به زیست باشند و نجات یابند به همان اندازه تمایل (سود) بیش‌تری به آن منطقه اختصاص می‌یابد. بنابراین موقعیتی که در آن بیش‌ترین تعداد تخم‌ها نجات یابند، پارامتری خواهد بود که الگوریتم بهینه‌سازی فاخته قصد بهینه‌سازی آن را دارد [۲۰، ۲۱].

پس از چند تکرار، تمام جمعیت فاخته‌ها به یک نقطه بهینه با حداکثر شباهت تخم‌ها به تخم‌های پرندگان میزبان و همچنین به محل بیش‌ترین منابع غذایی می‌رسند. این محل بیش‌ترین سود کلی را خواهد داشت و در آن کم‌ترین تعداد تخم‌ها از بین خواهد رفت.

گام‌های اساسی جستجوی فاخته به شرح ذیل است:

۱. هر فاخته هر بار فقط یک تخم (جواب) می‌گذارد و آن را درون لانه‌ای که به‌تصادف انتخاب می‌کند، قرار می‌دهد.
۲. لانه‌هایی که حاوی جواب‌های بهتر هستند به مرحله بعد منتقل می‌شوند.
۳. تعداد لانه‌های میزبان ثابت است و پرنده میزبان با احتمال Pa قادر به شناسایی جواب بیگانه خواهد بود. در این مورد پرنده میزبان می‌تواند تخم را دور انداخته و یا لانه را ترک کرده لانه‌ای جدید برای خود بسازد.

برای سادگی بیش‌تر، فرض آخر را می‌توان با جانشینی P_a درصد از n لانه موجود با لانه‌های جدید (جواب‌های تصادفی جدید) شبیه‌سازی کرد.

۳-۴- پرواز لوی

پرواز لوی یک قدم زدن تصادفی است که در آن طول گام‌ها از توزیع لوی پیروی می‌کند. توزیع لوی که توسط پائول پیر لوی[۱۹] ارائه شد، یکی از معدود توزیع‌هایی است که پایدار بوده و دارای توابع چگالی احتمال تحلیلی می‌باشد [۲۰]. به‌منظور پیاده‌سازی و استفاده از این توزیع، تولید اعداد تصادفی با پرواز لوی از دو مرحله تشکیل می‌شود که شامل انتخاب جهت تصادفی و تولید اندازه هر گام که از توزیع لوی پیروی کند، است. تولید جهت می‌تواند از توزیع یکنواخت باشد. برای تولید گام از توزیع لوی، چند راه وجود دارد. یکی از راه‌های کارآمد و آسان استفاده از الگوریتم مانتگنا[۲۰] برای توزیع پایدار و متقارن لوی است. البته منظور از متقارن این است که طول گام‌ها می‌تواند مثبت یا منفی باشد [۲۲].

مطالعات نشان داده است که پرواز لوی می‌تواند کارایی فرآیند جستجو در شرایط عدم اطمینان را بیشینه نماید [۲۳]. درواقع پرواز لوی در رفتار حیواناتی نظیر کرم میوه، نوعی مرغابی و میمون‌های عنکبوتی در جستجوی غذا، مشاهده شده است. به‌علاوه توزیع لوی کاربردهای زیادی در پدیده‌های فیزیکی مانند انتشار مولکولی فلوروسنت تحت شرایط مطلوب دارد.

تولید جواب تصادفی $X(t+1)$ بدین صورت است، که ابتدا اعداد تصادفی را از توزیع احتمال لوی تولید کرده، سپس در فضای جواب‌های شدنی، جوابی را با استفاده از آن عدد تصادفی همانند معادله ۵ تولید می‌نمائیم.

$$X(t+1) = \alpha \times Levy(\beta) \times (X(t) - best) \qquad (۵)$$

که در آن $\alpha>0$ طول گام بوده که متناسب با مقیاس مسئله و اندازه فضای جواب انتخاب می‌شود و در بیش‌تر موارد نیز می‌توان آن را مساوی یک قرار داد. رابطه بالا در اصل یک رابطه تصادفی برای قدم زدن یا پرواز تصادفی است. درواقع یک زنجیره مارکوف است که جواب بعدی در آن فقط به جواب جاری بستگی دارد. برخی از جواب‌های جدید با پرواز لوی در همسایگی بهترین جوابی (best) که در هر مرحله به دست می‌آید انتخاب می‌شوند، تا جستجوی محلی سرعت بیش‌تری داشته باشد. همچنین، بخشی از جواب‌های جدید را باید در میدانی دورتر از بهترین جواب‌های جاری به تصادف انتخاب کرد، تا فرآیند جستجو در دام بهینه‌های محلی نیفتد.

۴- روش پیشنهادی

در مسئله زمان‌بندی کارکارگاهی انعطاف‌پذیر هدف کاهش زمان اتمام کل کارها می‌باشد. برای رسیدن به این هدف، الگوریتم پیشنهادی در سه مرحله اجرا می‌شود.

در ابتدا برای بهبود کیفیت راه‌حل، توالی کارها و ترتیب ورود جمعیت اولیه براساس الگوریتم NEH-D تعیین شده است، الگوریتم NEH-D توسعه‌یافته الگوریتم NEH می‌باشد که بر اساس انحراف معیار بهبود یافته است و بر طبق کاهش زمان اجرای هر یک از کارها در تمام مراحل تعیین می‌شود. در مرحله دوم خوشه‌بندی مارکوف روی ماشین‌ها صورت می‌گیرد. با این دید که در هر مرحله از عملیات، کارها به تعدادی از ماشین‌ها تخصیص می‌یابد و تعدادی از ماشین‌ها فعال می‌باشند. بنابراین جهت بهینه‌کردن ورودی الگوریتم فاخته، خوشه‌بندی مارکوف انجام می‌شود، تا در هر مرحله از عملیات انتخاب ماشین از بین ماشین‌های فعال صورت گیرد. بنابراین تعداد جواب‌های انتخابی برای الگوریتم جستجوی فاخته محدود می‌گردد، تا سرعت اجرای الگوریتم فاخته افزایش یابد. درنهایت نیز الگوریتم جستجوی فاخته مطابق الگوریتم ۱ اجرا می‌گردد.

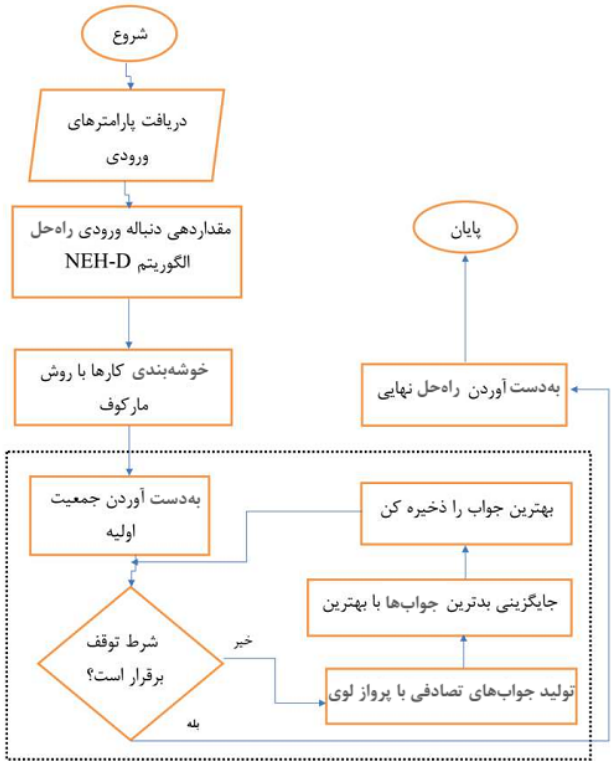
شروع: تابع هدف برابر است با $Min\ f(X)$ و $X=(x_1,...,x_d)^T$
۱- لانه میزبان $X_i\ (i=1,2,...,n)$ را به‌عنوان جمعیت اولیه تولید کن.
۲- تا زمانی‌که $(t<MaxGeneration)$ یا (معیار توقف)
۲-۱- یک جواب جدید تصادفی با پرواز لوی تولید کرده برازندگی‌اش (f_i) را به‌دست آور.
۲-۲- یک لانه به تصادف انتخاب کن (برای مثال لانه j)
اگر $(f_i>f_j)$ j را با جواب جدید جابجا کن
۲-۳- درصدی از بدترین لانه‌ها را رها کرده لانه‌های جدید درست کن.
۲-۴- بهترین جواب مرحله جاری را پیدا کن و ذخیره کن.

الگوریتم ۱: الگوریتم جستجوی فاخته

در این مرحله جواب اولیه تولید می‌شود و تابع هدف آن محاسبه می‌شود، سپس برخی از جواب‌های جدید با پرواز لوی در همسایگی بهترین جواب‌ها انتخاب می‌شوند، تا جستجوی محلی سرعت بیش‌تری داشته باشد. همچنین، بخشی از جواب‌های جدید را باید در میدانی دورتر از بهترین جواب‌ها به‌تصادف انتخاب کرد تا فرایند جستجو در دام بهینه‌های محلی نیفتد. حال جواب‌های تصادفی با پرواز لوی انتخاب و با جواب‌های جاری مقایسه می‌شود. اگر جواب‌های جدید بهتر باشند، جایگزین جواب‌های قبلی در لانه می‌شوند. این جواب‌های جدید را می‌توان با پرواز لوی در همسایگی لانه‌های انتخاب‌شده و یا در کل فضای مسئله به دست آورد. ما در هر مرحله برای تولید جواب جدید نسبتی تصادفی از فاصله هر لانه تا بهترین جواب جاری را به جواب جاری در لانه انتخابی اضافه می‌کنیم. با این کار درواقع با حفظ بهترین جواب جاری، سایر جواب‌ها به‌سمت بهترین جواب در حرکت هستند. پس از مقایسه و جانشینی بهترین‌ها، نسل جدید جواب‌ها به وجود می‌آیند که از میان آن‌ها بهترین جواب انتخاب و به‌عنوان بهترین جواب جاری جدید معرفی می‌شود.

تا اینجا یک تکرار از الگوریتم اجرا شده است. برای ادامه کار دوباره جواب‌های جدید تولید و مراحل بالا در الگوریتم فاخته تکرار می‌شود. این مرحله تا تخصیص تمام کارها و با توجه به پارامترهای مسئله تکرار

می‌گردد. فلوچارت مراحل مختلف روش پیشنهادی در شکل ۱ مشاهده می‌شود.



شکل ۱: فلوچارت الگوریتم پیشنهادی

۵- آزمایش‌ها

در این بخش نتایج حاصل از پیاده‌سازی الگوریتم پیشنهادی و مقایسه کارایی آن در برابر الگوریتم‌های دیگر ارائه شده است. الگوریتم پیشنهادی با الگوریتم‌هایی که در سال‌های اخیر برای مسئله زمان‌بندی تولید کارگاهی انعطاف‌پذیر ارائه شده‌اند، مقایسه شده است.

در آزمایش کارایی الگوریتم پیشنهادی از سه مجموعه داده آزمایش استاندارد Kacem [۲۴، ۲۵] و Brandimarte [۲۶] و مجموعه داده‌های مقاله [۲۷] استفاده شده است. محدوده تعداد کارها و ماشین‌ها در مجموعه داده‌های مقاله [۲۷] از ۲×۲ تا ۱۲×۸، در مجموعه داده آزمایش Kacem از ۴×۵ تا ۱۰×۱۵ و در مجموعه داده آزمایش Brandimarte از ۱۰×۶ تا ۲۰×۱۰ متغیر می‌باشد تا بتوانند فضای مسئله کوچک، متوسط و بزرگ را پوشش دهند. سه تابع هدف به‌صورت زیر تعریف شده است:

$$\begin{aligned} f1 &: makespan = \max\left\{C_i \,|\, i = 1,2,..., n\right\} \\ f2 &: total workload = \sum P_{i,j,k} \\ f3 &: \max\ workload = \max\left\{W_k \,|\, k = 1,2,.., m\right\} \end{aligned} \qquad (۶)$$

f1: حداکثر زمان تکمیل کارها

f2: زمان کاری همه ماشین‌ها

f3: بار کاری ماشین بحرانی

که هدف کمینه کردن توابع فوق می‌باشد.

مجموعه داده آزمایشی Kacem [۲۴، ۲۵] شامل پنج نمونه داده است. در این مجموعه، الگوریتم پیشنهادی با الگوریتم‌هایی شامل AL+CGA [۲۵]، PSO+SA [۲۷]، MOEA-GLS [۲۸]، PSO+TS [۲۹]، X-LS [۳۰]، X-SM [۳۱]، AIA [۳۲]، MOPSOP+LS [۲۰]، MOGA [۲۱]، P-DABC [۲۲]، PLS [۳۳]، HSFL [۳۴]، H-MOEA [۳۵] و HICA [۳۶] مقایسه شده است. همان‌طور که در جدول ۱ مشاهده می‌شود، الگوریتم پیشنهادی نسبت به الگوریتم‌های دیگر در مواردی کارایی بهتری را از خود نشان داده است. با بررسی نتایج به‌دست آمده در جدول ۱ می‌توان مشاهده کرد که راه‌حل الگوریتم پیشنهادی برای نمونه داده ۸×۸ بهتر از راه‌حل‌های الگوریتم AL+CGA، PSO+SA، MOGA و H-MOE است. همچنین برای نمونه داده ۷×۱۰ در این جدول، الگوریتم پیشنهادی راه‌حل بهتری از الگوریتم P-DABC ارائه کرده است.

الگوریتم HICA عملکرد مشابهی با الگوریتم پیشنهادی از خود نشان می‌دهد و راه‌حل‌های ارائه‌شده توسط هر دو الگوریتم یکسان است.

مجموعه داده آزمایش Brandimarte، شامل چهار مجموعه داده آزمایش با نام‌های ۶×۱۰(MK01)، ۶×۱۰(MK02)، ۸×۱ (MK03) و ۱۰×۲۰(MK08) است. در این مجموعه داده آزمایش الگوریتم پیشنهادی با الگوریتم‌های ابتکاری X-LS [۳۰]، AIA [۳۲]، PLS [۳۳]، HSFLA [۳۴] و HICA [۳۶] مقایسه شده است.

جدول ۲ نتایج حاصل از اجرای الگوریتم پیشنهادی بر روی این مجموعه داده آزمایش را نشان می‌دهد. همان‌طور که مشاهده می‌شود الگوریتم پیشنهادی می‌تواند برای هر نمونه داده آزمایش چندین راه‌حل بهینه ارائه دهد، که نشان‌دهنده قدرت بالای الگوریتم در جستجوی نقاط مختلف مسئله است. راه‌حل الگوریتم پیشنهادی بهتر از راه‌حل الگوریتم‌های X-LS و AIA است. همچنین در این جدول، الگوریتم پیشنهادی راه‌حل‌های مشابه با الگوریتم HICA ارائه کرده است.

در مقایسه با تعدادی از روش‌های اخیر ازجمله [۱۲، ۱۳]، که مسئله زمان‌بندی کار کارگاهی را به‌صورت تک‌هدفه بررسی کرده‌اند، نیز نتایج روش پیشنهادی قابل‌قبول است. برای مثال در [۱۲] که از بهبود الگوریتم PSO استفاده کرده است، زمان تکمیل کارها در داده‌های Kacem و مسائل ۸×۸، ۱۰×۱۰ و ۱۵×۱۰ به ترتیب ۱۴، ۷ و ۱۱ به دست آمده است، که عینا با نتایج این پژوهش متناظر است. همچنین [۱۲] بر روی مسائل MK01، MK02، MK03 و MK08 از داده‌های Brandimarte به ترتیب به زمان کل ۳۷، ۲۶، ۲۰۴ و ۵۲۳ رسیده است. مرجع [۱۳] نیز ارزیابی‌های خود را بر روی مسائل MK01، MK02، MK03 و MK08 از داده‌های Brandimarte انجام داده است و نتایج حاصل از آن برای معیار زمان تکمیل کارها به ترتیب ۳۶، ۲۴، ۲۰۴ و ۵۲۳ بوده است، که با نتایج حاصل از روش پیشنهادی قابل مقایسه است. نکته مهم این است که مقایسات نشان می‌دهد، روش پیشنهادی

قابلیت بهتری در حل مسائل بزرگ‌تر مانند MK03 و MK08 دارد. البته باید در نظر داشت که در مسائل چندهدفه ما چندین تابع را به‌صورت همزمان بهینه می‌کنیم و این به‌مراتب پیچیده تر از بهینه‌سازی یک تابع است.

جدول ۱: مقایسه نتایج الگوریتم‌ها بر روی داده آزمایش Kacem

Algorithm	۸×۸			۱۰×۱۰			۱۵×۱۰		
	f1	*f2*	*f3*	*f1*	*f2*	*f3*	*f1*	*f2*	*f3*
AL+CGA [۲۵]	۱۵	۷۹	۱۳	۷	۴۵	۵	۲۳	۹۵	۱۱
	۱۶	۷۵	۱۳				۲۴	۹۱	۱۱
PSO+SA [۳۹]	۱۵	۷۵	۱۲	۷	۴۴	۶	۱۲	۹۱	۱۱
	۱۶	۷۳	۱۳						
PSO+TS [۲۹]	۱۴	۷۷	۱۲	۷	۴۳	۶	۱۱	۹۳	۱۱
X-SM [۳۱]	۱۴	۷۷	۱۲	۷	۴۲	۶	۱۱	۹۱	۱۱
AIA [۳۲]	۱۴	۷۷	۱۲	۷	۴۳	۵	۱۱	۹۳	۱۱
MOGA [۲۱]	۱۵	۸۱	۱۱	۸	۴۲	۵	۱۱	۹۱	۱۱
	۱۵	۷۵	۱۲	۷	۴۲	۶	۱۲	۹۵	۱۰
P-DABC [۲۲]	۱۴	۷۷	۱۲	۸	۴۱	۷	۱۲	۹۱	۱۱
	۱۵	۷۵	۱۲	۷	۴۳	۵	۱۱	۹۳	۱۱
MOPSOP+LS [۲۰]	۱۵	۷۵	۱۲	۸	۴۱	۷	۱۲	۹۳	۱۰
	۱۴	۷۷	۱۲	۷	۴۲	۶	۱۱	۹۱	۱۱
HSFLA [۳۴]	۱۴	۷۷	۱۲	۷	۴۳	۵	۱۱	۹۱	۱۱
	۱۵	۷۵	۱۲	۸	۴۲	۵	۱۱	۹۳	۱۰
H-MOEA [۳۵]	۱۶	۷۳	۱۳	۸	۴۱	۷	۱۱	۹۳	۱۰
	۱۵	۷۵	۱۲	۷	۴۲	۶	۱۱	۹۱	۱۱
HICA [۳۶]	۱۴	۷۷	۱۲	۷	۴۲	۶	۱۱	۹۱	۱۱
	۱۵	۷۵	۱۲	۷	۴۳	۵	۱۱	۹۳	۱۰
Proposed Algorithm	۱۴	۷۵	۱۴	۷	۴۲	۷	۱۱	۹۱	۱۱
	۱۵	۷۵	۱۵	۸	۴۲	۸			

جدول ۲: مقایسه نتایج الگوریتم پیشنهادی بر روی داده‌های Brandimarte

Algorithm	MK01 (۱۰×۶)			MK02 (۱۰×۶)			MK03 (۱۵×۸)			MK08 (۲۰×۱۰)		
	f1	*f2*	*f3*	*f1*	*f2*	*f3*	*f1*	*f2*	*f3*	*f1*	*f2*	*f3*
X-LS [۳۰]	۴۲	۱۶۲	۴۲	۲۸	۱۵۵	۲۸	۲۰۴	۸۵۲	۲۰۴	۵۲۳	۲۵۲۴	۵۲۳
AIA [۳۲]	۴۰	۱۷۱	۳۶	۲۶	۱۵۴	۲۶	۲۰۴	۱۲۰۷	۲۰۴	۵۲۳	۲۷۲۳	۵۲۳
PLS [۳۳]	۴۰	۱۶۷	۳۶	۲۶	۱۵۱	۲۶	۲۰۴	۸۵۲	۲۰۴	۵۲۳	۲۵۲۴	۵۲۳
	۴۱	۱۶۰	۳۹	۲۷	۱۴۶	۲۷	۲۱۰	۸۵۰	۲۰۴	۵۲۴	۲۵۱۹	۵۲۴
	۴۲	۱۶۳	۳۷	۲۸	۱۴۵	۲۸	۲۱۳	۸۴۶	۲۱۳	۵۳۳	۲۵۱۴	۵۳۳
HSFLA [۳۴]	۴۰	۱۶۷	۳۶	۲۶	۱۵۲	۲۶	۲۰۴	۸۵۲	۲۰۴	۵۲۳	۲۵۲۴	۵۲۳
	۴۰	۱۶۵	۳۷	۲۷	۱۵۱	۲۶	۲۱۰	۸۵۰	۲۰۴	۵۲۴	۲۵۱۹	۵۲۴
	۴۱	۱۶۴	۴۰	۲۷	۱۵۰	۲۷	۲۱۳	۸۴۶	۲۱۳	۵۳۳	۲۵۱۴	۵۳۳
	۴۲	۱۶۳	۴۲	۲۸	۱۴۶	۲۸	۲۲۰	۸۴۸	۲۱۰	۵۴۸	۲۵۰۹	۵۴۲
HICA [۳۶]	۴۰	۱۶۵	۳۷	۲۶	۱۵۲	۲۶	۲۰۴	۸۵۲	۲۰۴	۵۲۳	۲۵۲۴	۵۲۳
	۴۰	۱۶۹	۳۶	۲۷	۱۵۰	۲۶	۲۱۰	۸۵۱	۲۱۰	۵۲۴	۲۵۱۹	۵۲۴
	۴۱	۱۶۰	۳۹	۲۹	۱۴۴	۲۸	۲۱۳	۸۴۴	۲۱۳	۵۳۳	۲۵۱۴	۵۳۳
Proposed Algorithm	۴۰	۱۶۵	۴۰	۲۶	۱۵۲	۲۶	۲۰۴	۸۵۲	۲۰۴	۵۲۳	۲۵۲۴	۵۲۳
	۴۱	۱۶۳	۴۱	۲۷	۱۵۰	۲۷	۲۱۰	۸۵۱	۲۱۰	۵۲۴	۲۵۱۹	۵۲۴
	۴۲	۱۶۰	۴۲	۲۸	۱۴۸	۲۸	۲۱۳	۸۴۵	۲۱۳	۵۳۳	۲۵۱۴	۵۳۳

مجموعه داده‌های مقاله [۲۷] بر اساس داده‌های واقعی یک شرکت مبل‌سازی که شامل بخش‌های مختلف است، جمع‌آوری شده است. در آزمایش‌ها بر روی این مجموعه داده، الگوریتم پیشنهادی با الگوریتم جستجوی فاخته استاندارد [۳۷] و الگوریتم رقابت استعماری [۳۸] مقایسه شده است. همچنین تأثیر انتخاب ترتیب ورود کارها به کمک NEH-D و خوشه‌بندی مارکوف در چهارچوب پیشنهادی نیز بررسی شده است. در ادامه و در جدول‌های ۳ و ۴، CS اشاره به الگوریتم جستجوی فاخته استاندارد [۳۷] دارد، ICA الگوریتم رقابت استعماری استاندارد [۳۸] است، ICA-Markov اشاره به الگوریتم رقابت استعماری با خوشه‌بندی مارکوف و ICA-NEH-Markov اشاره به الگوریتم رقابت استعماری با ورودی NEH-D و خوشه‌بندی مارکوف دارد. همچنین ستون اندازه به ترتیب: تعداد عملیات، تعداد ماشین‌ها و تعداد کارها را نشان می‌دهند. به‌طورمثال ۳-۵-۴ یعنی ۴ کار روی ۵ ماشین و در ۳ مرحله عملیات است.

جدول ۳: مقایسه تابع هدف حداکثر زمان تکمیل کارها (makespan) در الگوریتم‌های مختلف

نام	اندازه	CS[۳۷]	ICA [۳۸]	ICA-Markov	ICA-NEH-Markov	Proposed Algorithm
SFJS1	۲-۲-۲	۶۸±۱	۶۸±۱	۶۶±۰	۶۶±۰	۶۶±۰
SFJS3	۲-۲-۳	۲۲۱±۰	۲۵۹±۱	۲۳۷±۱	۲۲۱±۰	۲۲۱±۰
SFJS4	۲-۲-۳	۳۶۷±۲	۳۵۵±۰	۳۵۵±۰	۳۵۵±۰	۳۵۵±۱
SFJS5	۲-۲-۳	۱۳۰±۲	۱۳۴±۱	۱۳۱±۳	۱۳۰±۱	۱۲۸±۰
SFJS6	۳-۳-۳	۳۲۷±۳	۳۳۰±۳	۳۲۸±۱	۳۲۷±۱	۳۲۰±۲
SFJS7	۳-۵-۳	۳۲۲±۴	۳۴۵±۳	۳۴۳±۱	۳۳۰±۱	۳۲۰±۱
SFJS8	۳-۴-۳	۲۵۳±۲	۲۸۵±۲	۲۸۳±۱	۲۶۴±۱	۲۵۳±۰
SFJS9	۳-۳-۳	۲۲۰±۳	۲۳۰±۲	۲۲۵±۳	۲۱۰±۰	۲۱۰±۰
SFJS10	۴-۵-۳	۵۵۰±۳	۵۷۸±۳	۵۶۷±۳	۵۵۳±۱	۵۵۰±۱
MFJS1	۵-۶-۳	۵۰۰±۴	۵۷۸±۴	۵۶۰±۲	۵۲۱±۲	۴۷۷±۱
MFJS2	۵-۷-۳	۴۹۷±۶	۵۶۳±۵	۵۶۰±۳	۵۶۰±۲	۴۵۷±۳
MFJS3	۶-۷-۳	۵۵۴±۵	۶۴۵±۴	۶۴۵±۳	۶۴۳±۲	۵۲۱±۳
MFJS4	۷-۷-۳	۶۵۷±۵	۶۷۰±۵	۶۶۷±۳	۶۶۳±۳	۶۴۸±۳
MFJS5	۷-۷-۳	۶۴۵±۶	۷۰۶±۴	۷۰۵±۵	۷۰۴±۳	۶۲۵±۲
MFJS6	۸-۷-۳	۷۴۶±۶	۷۸۷±۴	۷۸۲±۴	۷۶۵±۲	۷۲۰±۲
MFJS8	۹-۸-۴	۱۰۹۰±۵	۱۲۱۲±۶	۱۲۰۵±۳	۱۱۷۴±۳	۱۰۴۴±۳
MFJS9	۱۱-۸-۴	۱۳۵۱±۶	۱۵۱۶±۶	۱۵۰۲±۴	۱۴۵۱±۳	۱۳۴۲±۳
MFJS10	۱۲-۸-۴	۱۶۰۷±۶	۱۶۸۷±۵	۱۶۳۴±۷	۱۶۰۵±۳	۱۵۷۲±۳

در آزمایش‌های جدول ۳، تأثیر رویکردهای مختلف پیشنهادی در رسیدن به پاسخ بهینه و کاهش حداکثر زمان تکمیل کارها (makespan) به‌عنوان تابع هدف اصلی، نشان داده شده است. همان‌طور که جدول ۳ نشان می‌دهد، الگوریتم پیشنهادی از لحاظ زمان اتمام کل، دارای جواب بهتری نسبت به دیگر الگوریتم‌ها است. همچنین علی‌رغم اینکه استفاده از خوشه‌بندی مارکوف (ستون پنجم) اندکی در بهبود نتایج نسبت به روش‌های پایه ازجمله روش [۳۸] مؤثر بوده است، اما تأثیر استفاده از الگوریتم NEH-D (ستون ششم) بیش‌تر بوده است و نتایج بهتری ارائه داده است. همچنین استفاده از پرواز لوی در روش

پیشنهادی در کنار دو رویکرد دیگر نتایج را در ستون آخر جدول ۳ به شکل چشم‌گیری بهبود داده است. البته تأثیر اندک خوشه‌بندی مارکوف از ابتدا نیز قابل پیش‌بینی بود، زیرا مهم‌ترین هدف استفاده از خوشه‌بندی مارکوف کاهش فضای جستجو و زمان اجرا است و این کار به تنهایی به پاسخ‌های بهتر منتهی نمی‌شود.

جدول ۴: مقایسه دفعات اجرای تابع هدف در الگوریتم‌های مختلف

نام	اندازه	[۳۷] CS	[۳۸] ICA	ICA-Markov	Proposed Algorithm
SFJS1	۲-۲-۲	۵۰	۵۰	۵۰	۵۰
SFJS3	۳-۲-۲	۵۵۲	۵۵۲	۵۵۰	۵۵۰
SFJS4	۳-۲-۲	۵۰۲	۵۰۴	۵۰۰	۵۰۰
SFJS5	۳-۲-۲	۵۰۰	۵۰۰	۵۰۰	۵۰۰
SFJS6	۳-۳-۳	۱۳۷۱	۱۴۷۴	۱۲۵۰	۱۲۵۰
SFJS7	۳-۵-۳	۱۵۳۴	۱۸۲۴	۱۲۵۰	۱۲۵۰
SFJS8	۳-۴-۳	۱۵۰۱	۱۷۱۷	۱۲۵۰	۱۲۵۰
SFJS9	۳-۳-۳	۱۴۰۳	۱۵۱۴	۱۲۵۰	۱۲۵۰
SFJS10	۴-۵-۳	۱۴۸۸	۱۷۴۱	۱۲۵۰	۱۲۵۰
MFJS1	۵-۶-۳	۱۶۹۱	۲۱۳۰	۱۲۵۰	۱۲۵۰
MFJS2	۵-۷-۳	۷۰۱۹	۸۱۴۱	۵۷۵۰	۵۷۵۰
MFJS3	۶-۷-۳	۷۱۲۰	۸۶۲۹	۵۷۵۰	۵۷۵۰
MFJS4	۷-۷-۳	۱۱۱۳۲	۱۳۱۴۶	۹۱۵۰	۹۱۵۰
MFJS5	۷-۷-۳	۱۱۱۶۷	۱۳۱۴۲	۹۱۵۰	۹۱۵۰
MFJS6	۸-۷-۳	۱۱۴۱۷	۱۳۶۲۰	۹۱۵۰	۹۱۵۰
MFJS8	۹-۸-۴	۶۴۱۳۹	۷۷۸۲۰	۵۰۰۵۰	۵۰۰۵۰
MFJS9	۱۱-۸-۴	۱۲۴۱۲۲	۱۴۶۳۶۴	۱۰۲۰۰۰	۱۰۲۰۰۰
MFJS10	۱۲-۸-۴	۱۳۰۰۱۳	۱۶۳۸۱۳	۱۰۲۰۰۰	۱۰۲۰۰۰

در جدول ۴ تعداد دفعات ارزیابی تابع هدف توسط الگوریتم پیشنهادی با وجود و بدون وجود خوشه‌بندی مارکوف نشان داده شده است. هدف از این آزمایش‌ها بررسی تأثیر رویکرد پیشنهادی در تسریع همگرایی است. همان‌طور که انتظار داشتیم، تأثیر خوشه‌بندی مارکوف در همگرایی به پاسخ نهایی کاملاً مشخص بوده و در الگوریتم با خوشه‌بندی مارکوف، تعداد دفعات اجرای تابع هدف به نسبت دیگر الگوریتم‌ها کم‌تر بوده است. ازآنجاکه شرایط یکسانی برای تمامی الگوریتم‌ها در نظر گرفته شده است، در الگوریتم‌های بدون خوشه‌بندی دفعات محاسبه تابع هدف بیش‌تر از الگوریتم‌های با خوشه‌بندی مارکوف می‌باشد و محاسبات بیش‌تری انجام می‌دهد، ازاین‌رو تعداد جواب‌های تولیدشده بیش‌تر بوده است. بنابراین تأثیر خوشه‌بندی مارکوف کاملاً آشکار است. همچنین ستون آخر که به روش پیشنهادی اشاره دارد، از دو رویکرد دیگر NEH-D و پرواز لوی استفاده کرده است. با وجود استفاده از این دو رویکرد هیچ‌گونه تأثیری بر روی تعداد دفعات ارزیابی تابع هدف مشاهده نمی‌شود. از این‌رو از بررسی تأثیر جداگانه هر یک از این دو رویکرد در این زمینه صرف‌نظر شده است.

در مجموع، با جمع‌بندی جدول‌های ۳ و ۴ می‌توان نتیجه گرفت که دو رویکرد الگوریتم NEH-D و پرواز لوی تاثیر قابل‌توجهی در رسیدن به پاسخ‌های بهینه دارند و در مقابل مهم‌ترین تأثیر خوشه‌بندی مارکوف در کاهش زمان اجرای الگوریتم و تسریع همگرایی است.

۶- نتیجه‌گیری

در زمان‌بندی کار کارگاهی انعطاف‌پذیر، هدف تخصیص مجموعه‌ای از کارها به مجموعه‌ای از ماشین‌ها با شرایط خاص است. در مسئله موردبررسی، هدف ما کاهش زمان اتمام کل کارها است و برای رسیدن به این هدف در ابتدا الگوریتم NEH-D جهت ترتیب ورود اولیه کارها اجرا می‌گردد. سپس ماشین‌های فعال با روش مارکوف خوشه‌بندی می‌گردند تا ورودی الگوریتم فاخته محدود گردد و درنهایت نیز الگوریتم جستجوی فاخته با استفاده از پرواز لوی اجرا می‌گردد.

این استراتژی بهینه‌سازی و کارا و نیرومند را به‌سادگی می‌توان برای مسائل بهینه‌سازی با توابع هدف چندگانه و محدودیت‌های مختلف و همچنین مسائل دشوار با زمان حل غیرچندجمله‌ای تعمیم داد و این الگوریتم قابل تعمیم به سایر مسائل چندهدفه نیز است. حداقل کردن زمان کل کارها تنها بخش کوچکی از مسائل زمان‌بندی می‌باشد و می‌توان بر روی مسائل حداقل بیکاری ماشین‌ها، وابستگی بین کارها، سرعت فرآیند جستجو و مسایلی از این دست نیز متمرکز شد. با توجه به شرایط مسائل زمان‌بندی به‌نظر می‌رسد، ادغام الگوریتم جاری با الگوریتم‌های تکاملی دیگر پیشنهاد خوبی برای تحقیقات بعدی باشد و منجر به روش‌های بهتری شود. مهم‌تر از همه، مانند اغلب الگوریتم‌های فراابتکاری، نیاز به تحلیل ریاضی ساختارهای الگوریتم به‌شدت احساس می‌شود و تاکنون هیچ چهارچوب ریاضی مناسبی برای الگوریتم‌های فراابتکاری ارائه نشده است. هرگونه پیشرفت در این زمینه به درک چگونگی سازوکار الگوریتم‌های فراابتکاری کمک شایانی خواهد کرد.

مراجع

[1] K. R. Baker, *Elements of Sequencing and Scheduling*, Dartmouth College, Amos Tuck School of Busines,1998.

[2] D. Gupta, and T. Magnusson, "The capacitated lot-sizing and scheduling problem with sequence-dependent setup costs and setup times," *Computers and Operations Research*, vol. 32, no. 4, pp. 727-747, 2005.

[3] P. Brucker, and R. Schlie, "Job-shop scheduling with multi-purpose machines," *Computing*, vol. 45, no. 4, pp. 369-375, 1990.

[4] L. N. Xing, Y. W. Chen, P. Wang, Q. S. Zhao, and J. Xiong, "A knowledge-based ant colony optimization for flexible job shop scheduling problems," *Applied Soft Computing*, vol. 10, no. 3, pp. 888-896, 2010.

[5] W. Teekeng, and A. Thammano, "A combination of shuffled frog leaping and fuzzy logic for flexible job-shop scheduling problems," *Procedia Computer Science*, vol. 6, no. 4, pp. 69-75, 2011.

[6] J. Zhang, P. Zhang, J. X. Yang, and Y. Huang, "Solving the job shop scheduling problem using the imperialist competitive algorithm," *Advanced Materials Research*, vol. 430-432, pp. 737-740, 2012.

[7] F. Jolai, M. Rabiee, and H. Asefi, "A novel hybrid meta-heuristic algorithm for a no-wait flexible flow shop scheduling problem with sequence dependent setup times," *International Journal of Production Research*, vol. 50, no. 24, pp. 7447-7466, 2012.

[8] S. Burnwal, and S. Deb, "Scheduling optimization of flexible manufacturing system using cuckoo search-based approach," *The International Journal of Advanced Manufacturing Technology*, vol. 64, no. 5, pp. 951-959, 2013.

[9] R. Babukartik, and P. Dhavachelvan, "Hybrid algorithm using the advantage of ACO and Cuckoo search for job scheduling," *International Journal of Information Technology Convergence and Services*, vol. 2, no. 4, pp. 25-34, 2012.

[10] A. T. S. Al-Obaidi, and A. B. A. D. Majeed, "Proposal of tabu search algorithm based on cuckoo search," *International Journal of Advanced Research in Artificial Intelligence (IJARAI)*, vol. 3, no. 3, 2014.

[11] J. Lin, "A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem," *Knowledge-Based Systems*, vol. 78, pp. 59-74, 2015.

[12] M. R. Singh, and S. S. Mahapatra, "A quantum behaved particle swarm optimization for flexible job shop scheduling," *Computers & Industrial Engineering*, vol. 93, pp. 36-44, 2016.

[13] S. Ishikawa, R. Kubota, and Keiichi Horio, "Effective hierarchical optimization by a hierarchical multi-space competitive genetic algorithm for the flexible job-shop scheduling problem," *Expert Systems with Applications*, vol. 42, no. 24, pp. 9434-9440, 2015.

[14] M. Marichelvam, T. Prabaharan, and X. S. Yang, "Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan," *Applied Soft Computing*, vol. 19, pp. 93-101, 2014.

[15] M. Ebrahimi, S. M. T. Fatemi Ghomi, and B. Karimi, "Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates," *Applied Mathematical Modelling*, vol. 38, pp. 2490-2504, 2014.

[۱۶]نبی طاهری، رحمت‌الله هوشمند و رضا همتی، «برنامه‌ریزی هماهنگ نصب منابع تولید پراکنده و توسعه شبکه توزیع در حضور نامعینی بار و قیمت انرژی،» *مجله مهندسی برق دانشگاه تبریز*، دوره ۴۴، شماره ۱، صفحه ۴۳-۵۶، ۱۳۹۳.

[۱۷]رحمت‌الله هوشمند، حسین محکمی و امین خدابخشیان، «روشی جدید در جایابی بهینه خازن‌ها و ژنراتورهای توزیع‌شده در شبکه توزیع با استفاده از الگوریتم جستجوی باکتری جهت‌داده‌شده با PSO،» *مجله مهندسی برق دانشگاه تبریز*، دوره ۳۹، شماره ۲، صفحه ۶۱-۷۲، ۱۳۸۸.

[18] L. Xiao-Ping, W. Yue-Xuan, and W. Cheng, "Heuristic algorithms for large flowshop scheduling problems," *5th World Congress on Intelligent Control and Automation (WCICA)*, vol. 4, pp. 2999-3003, 2004.

[19] D. Harel, and Y. Koren, "On clustering using random walks," *Foundations of Software Technology and Theoretical Computer Science (FST TCS)*, pp. 18-41, 2001.

[20] X. Wang, L. Gao, C. Zhang, and X. Shao, "A multi-objective genetic algorithm based on immune and entropy principle for flexible job-shop scheduling problem," *The International Journal of Advanced Manufacturing Technology*, vol. 51, no. 5, pp. 757-767, 2010.

[21] G. Moslehi, and M. Mahnam, "A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search," *International Journal of Production Economics*, vol. 129, no. 1, pp. 14-22, 2011.

[22] J. Q. Li, Q. K. Pan, and K. Z. Gao, "Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems," *The International Journal of Advanced Manufacturing Technology*, vol. 55, no. 12, pp. 1159-1169, 2011.

[23] X. S. Yang, and S. Deb,"Cuckoo search via Lévy flights," *World Congress on Nature & Biologically Inspired Computing (NaBIC)*, pp. 210-214, 2009.

[24] I. Kacem, S. Hammadi, and P. Borne, "Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic," *Mathematics and Computers in Simulation*, vol. 60, no. 3, pp. 245-276, 2002.

[25] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 32, no. 1, pp. 1-13, 2002.

[26] P. Brandimarte, "Routing and scheduling in a flexible job shop by tabu search," *Annals of Operations Research*, vol. 41, no. 3, pp. 157-183, 1993.

[27] P. Fattahi, M. S. Mehrabad, and F. Jolai, "Mathematical modeling and heuristic approaches to flexible job shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 18, no. 3, pp. 331-342, 2007.

[28] N. B. Ho, and J. C. Tay, "Solving multiple-objective flexible job shop problems by evolution and local search," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 38, no. 5, pp. 674-685, 2008.

[29] G. Zhang, X. Shao, P. Li, and L. Gao "An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem," *Computers and Industrial Engineering*, vol. 56, no. 4, pp. 1309-1318, 2009.

[30] L. N. Xing, Y. W. Chen, and K. W. Yang, "An efficient search method for multi-objective flexible job shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 20, no. 3, pp. 283-293, 2009.

[31] L. N. Xing, Y. W. Chen, and K. W. Yang, "Multi-objective flexible job shop schedule: design and evaluation by simulation modeling," *Applied Soft Computing*, vol. 9, no. 1, pp. 362-376, 2009.

[32] A. Bagheri, M. Zandieh, I. Mahdavia, and M. Yazdani "An artificial immune algorithm for the flexible job-shop scheduling problem," *Future Generation Computer Systems*, vol. 26, no. 4, pp. 533-541, 2010.

[33] J. Q. Li, Q. K. Pan, and J. Chen, "A hybrid Pareto-based local search algorithm for multi-objective flexible job shop scheduling problems," *International Journal of Production Research*, vol. 50, no. 4, pp. 1063-1078, 2012.

[34] J. Li, Q. Pan, and S. Xie, "An effective shuffled frog-leaping algorithm for multi-objective flexible job shop scheduling problems," *Applied Mathematics and Computation*, vol. 218, no. 18, pp. 9353-9371, 2012.

[35] J. Xiong *et al*., "A hybrid multiobjective evolutionary approach for flexible job-shop scheduling problems," *Mathematical Problems in Engineering*, 2012.

[۳۶]مرصاد شعبان‌پور، *یک روش جدید مبتنی بر الگوریتم رقابت استعماری برای حل مسئله زمان‌بندی کار کارگاهی انعطاف‌پذیر*، پایان‌نامه کارشناسی ارشد، دانشگاه آزاد اسلامی واحد مشهد، مشهد، ایران، ۱۳۹۲.

[37] R. Rajabioun, "Cuckoo optimization algorithm," *Applied Soft Computing*, vol. 11, no. 8, pp. 5508-5518, 2011.

[38] E. Atashpaz-Gargari, and C. Lucas. "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," *IEEE Congress on Evolutionary Computation*, pp. 4661-4667, 2007.

زیرنویس‌ها

۱ Flexible Job Shop
۲ Knowledge-Based Ant Colony Optimization (KBACO)
۳ Ant Colony Optimization(ACO)
۴ Biogeography-based Optimization (BBO)
۵ Particle Swarm Optimization (PSO)
۶ Logistic chaotic map
۷ Multi-space Competitive Distributed Genetic Algorithm (mcDGA)
۸ Sequence Dependent Family Setup Time (SDFST)
۹ Bacterial Foraging Algorithm
۱۰ Job Shop Scheduling Problem (JSSP)
۱۱ Partially Flexible Job Shop Scheduling Problem (PF-JSSP)
۱۲ Totally Flexible Job Shop Scheduling Problem (TF-JSSP)
۱۳ makespan
۱۴ total workload
۱۵ Nawaz, Enscore, and Ham (NEH)
۱۶ cuckoo search(CS)
۱۷ X. S. Yang
۱۸ S. Deb
۱۹ Paul Pierre Levy
۲۰ Mantegna