

# یک الگوریتم ترکیبی کارآمد بهبودیافته برای مسئله‌ی سفر چند فروشنده در مقیاس بزرگ

سید حمید میرمحمدی (دانشیار)

سیما امیری پبدنی (دانشجوی دکتری)

پریسا فیض الهی\* (دانشجوی دکتری)

دانشکده‌ی مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان

مهندسی صنایع و مدیریت شریف، زمستان ۱۴۰۰  
دوری ۱-۳۷، شماره ۲، ص. ۱۲۳-۱۳۳، (پادداشت ش)

مسئله‌ی چندین فروشنده‌ی دوره‌گرد (MTSP) گسترشی مشهور از مسئله‌ی فروشنده‌ی دوره‌گرد (TSP) است. تحقیقات این مسئله بر خلاف مسئله‌ی TSP که گستردگی آن توجه زیادی را به خود معطوف کرده است، بسیار محدود بوده و از این رو الگوریتم جدید ترکیبی موجود به نام الگوریتم ژنتیک - مورچگان بهبودیافته (IAC-PGA) ارائه شده است که در آن از یک روش جستجوی محلی به منظور بهبود الگوریتم بهره گرفته شده است. ایده‌ی اصلی این مقاله آن است که از الگوریتم ژنتیک برای تعیین تعداد شهرها و نقطه‌ی شروع هر فروشنده بهره بگیریم و سپس از الگوریتم مورچگان برای تعیین بهترین تور استفاده کنیم. نتایج حاصل از مقایسه‌ی نتایج الگوریتم با دیگر الگوریتم‌های موجود در ادبیات موضوع و تجزیه و تحلیل آن نشان می‌دهد که الگوریتم پیشنهادی در حل MTSP در مقیاس بزرگ مؤثر است.

واژگان کلیدی: الگوریتم ژنتیک، پارتو، الگوریتم کلونی مورچه‌ها، مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه همراه با الگوریتم ترکیبی بهبودیافته، روش جستجوی محلی ۲-opt.

h.mirmohammadi@iut.ac.ir  
eng.amiri940@gmail.com  
parisafeizollahy@in.iut.ac.ir

## ۱. مقدمه

برخی محققان در زمینه‌های مختلف به مسئله‌ی فروشنده‌ی دوره‌گرد (TSP) <sup>۱</sup> توجه کرده‌اند که یک مسئله‌ی کلاسیک NP-complete است که در بهینه‌سازی ترکیبی با آن روبه‌رو می‌شوند. <sup>[۱]</sup> هدف این است که با شرط این که همه گره‌های (شهرها) موجود فقط یک‌بار از طریق فروشنده بازدید شود، مسیر فروشنده با کمینه هزینه (مسافت) پیدا شود. برخی الگوریتم‌های دقیق و فراابتکاری می‌توانند TSP را به طور مؤثر حل کنند؛ همانند الگوریتم دقیق شاخه و برش <sup>[۲]</sup>، الگوریتم‌های تقریبی <sup>[۳]</sup>، الگوریتم ژنتیک <sup>[۴]</sup>، الگوریتم کلونی زنبورهای مصنوعی <sup>[۵]</sup> و <sup>[۶]</sup>، الگوریتم کلونی مورچه‌ها <sup>[۸]</sup>، الگوریتم چرخه‌ی آب <sup>[۹]</sup>، الگوریتم بهینه‌سازی ذرات <sup>[۱۱]</sup> و الگوریتم شبیه‌سازی تبرید. <sup>[۱۲]</sup>

مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه (MTSP) <sup>۲</sup> یک مدل کلی از مسئله‌ی مشهور فروشنده‌ی دوره‌گرد TSP است. در مورد MTSP، فروشندگان تعداد مشخصی از شهرها را به جای بازدید از کل شهرها (n) پیش رو دارند و تمام شهرها باید توسط مجموع فروشندگان بازدید شود و هدف این است که تورهایی برای همه‌ی فروشندگان (m) با کمینه مسافت پیدا شود. شهرهای شروع و خاتمه هر فروشنده تابع هدف هستیم.

مقاله به شرح زیر سازمان‌دهی شده است: بخش ۲ مفهوم و مرور ادبیات مسئله‌ی MTSP است. در بخش ۳ تعریف مسئله که شامل الگوریتم‌های ژنتیک پارتو (PGA) و الگوریتم کلونی مورچه‌ها (ACO) و همچنین یک الگوریتم بهبودیافته ژنتیک - مورچگان (IAC-PGA) <sup>۳</sup> است، بررسی می‌کند. همچنین روش حل MTSP در بخش ۳ معرفی شده است. در بخش ۴ نتایج محاسباتی الگوریتم‌های مختلف و آنالیز میان آنها مطرح شده و سرانجام بخش ۵ به نتیجه‌گیری مقاله و ایده‌های جدید به منظور مطالعات آینده می‌پردازد.

\* نویسنده مسئول

تاریخ: دریافت ۱۳۹۹/۳/۳۱، اصلاحیه ۱۳۹۹/۱۱/۲۵، پذیرش ۱۴۰۰/۴/۲۳.

DOI:10.24200/J65.2021.55500.2109

## ۲. مرور ادبیات

مدل MTSP به طور گسترده‌یی در برنامه‌ریزی<sup>[۱۷]</sup>، شبکه‌ی نگهداری و تعمیرات<sup>[۱۸]</sup>، تدارکات<sup>[۱۹]</sup> و بازرسی کیفیت به منظور انتخاب مسیر حمل و نقل<sup>[۲۰]</sup> به کار برده شده است.

در مقایسه با TSP استاندارد، تحقیق در مورد MTSP بسیار کم‌تر است. با عمیق شدن تحقیقات پژوهشگران در مسئله‌ی MTSP، این موضوع بیشتر مورد توجه قرار گرفته و از این رو بسیاری از روش‌های مؤثر توسعه یافته است. این رویکردها را می‌توان به سه نوع تقسیم کرد: رویکرد دقیق، رویکرد ابتکاری و فراابتکاری. الگوریتم‌های دقیق نوعی الگوریتم‌های اولیه مبتنی بر نظریه‌ی ریاضی‌اند. مشهورترین الگوریتم دقیق، روش شاخه و کران است که در ابتدا برای حل MTSP در مقیاس بزرگ پیشنهاد شده است.<sup>[۲۱]</sup> اگرچه الگوریتم‌های دقیق دارای خصوصیات ریاضی‌اند، اما توانایی حل مسئله‌ی آنها کاملاً به اندازه مسئله بستگی دارد. هنگامی که اندازه بزرگ می‌شود، ممکن است چنین مسئله‌یی در مدت زمان قابل قبول حل نشود، یا حتی قابل حل نباشد. برای غلبه بر مشکلات محاسباتی فوق، بیشتر محققان به تحقیق در مورد الگوریتم‌های اکتشافی می‌پردازند، که می‌تواند به راحتی یک راه حل بهینه یا نزدیک به بهینه برای MTSP با اندازه بزرگ به دست آورد. برخی از روش‌های دقیق و اکتشافی موجود MTSP شرح داده شده است.<sup>[۲۲]</sup> یک مسئله‌ی ناهمگن که به صورت چند نقطه‌ی شروع، مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه به یک مسئله‌ی فروشنده‌ی دوره‌گرد به صورت تکی و نامتقارن تبدیل شده و با استفاده از الگوریتم اکتشافی مشهور لین کرنیگان - هلسگا حل شده است.<sup>[۲۳]</sup> یک الگوریتم فراابتکاری بهبود یافته مبتنی بر الگوریتم کلونی زنبورهای مصنوعی<sup>[۲۴]</sup>، برای محدود کردن فضای جستجو و کاهش زمان محاسبه، برای چند فروشنده‌ی دوره‌گرد و چند نقطه‌ی شروع به صورت یکپارچه ارائه شده است. در همان سال، یک الگوریتم بهینه‌سازی ذرات و ژنتیک بهبود یافته برای MTSP به چاپ رسید که به منظور جلوگیری از عملیات تکراری در نقاط و کاهش مقدار محاسبه، الگوریتم ژنتیکی پارتو PGA برای MTSP طراحی شده است.<sup>[۲۵]</sup> یک مدل بهبود یافته با پارامترهای بهینه‌سازی درجه دو ارائه شده است که در بهینه‌سازی ساختار پیچیده‌ی توپولوژی استفاده می‌شود.<sup>[۲۶]</sup>

الگوریتم ژنتیک GA یک الگوریتم تکاملی است که اصول انتخابی آن شایسته‌ترین فرد میان جمعیت است.<sup>[۲۷]</sup> و برای مسئله‌ی MTSP توسعه‌ی الگوریتم‌های ژنتیکی (GA) برای کاهش فاصله‌ی کلی و تفاوت بین مسافت طی شده توسط هر فروشنده ارائه شده است.<sup>[۲۸]</sup> یک الگوریتم ژنتیک جدید مؤثر با اپراتورهای محلی در حل MTSP و تولید راه‌حل با کیفیت بالا در مدت زمان معقول برای برنامه‌های زندگی واقعی ارائه شده است.<sup>[۲۹]</sup> همان‌طور که دیده می‌شود، GA می‌تواند همه‌ی متغیرها را به طور هم‌زمان در نظر بگیرد (هم نقاط شروع و هم توالی شهرها)، اما به مقدار اولیه جمعیت بستگی دارد. وقتی فضای راه‌حل بزرگ‌تر می‌شود، توانایی الگوریتم برای یافتن راه‌حل بهینه بسیار ضعیف می‌شود. الگوریتم کلونی مورچه‌ها یک الگوریتم متداول برای MTSP است و ثابت شده است که یک روش قابل قبول برای تعدادی از مسائل Np-hard است.<sup>[۳۰]</sup> و این الگوریتم برای حل مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه تعیین شده مورد استفاده قرار گرفته است.<sup>[۳۱]</sup> مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه با وجود نقاط شروع ثابت با پارامترهای غیرتصادفی بررسی شده است.<sup>[۳۲]</sup> الگوریتم کلونی مورچه‌ها وابستگی ضعیفی به مقدار اولیه و توانایی عالی در جستجوی راه‌حل بهینه هنگام استفاده، برای حل مسئله‌ی MTSP نشان می‌دهد. با این وجود، نقاط شروع فروشنده را تعیین

نمی‌کند. در نتیجه، همیشه برای حل MTSP با نقاط شروع ثابت استفاده می‌شود. با توجه به این که MTSP از نظر تعداد متغیرهای موجود دارای پیچیدگی بالاتری نسبت به فروشنده‌ی دوره‌گرد است. دست‌یابی رویکردهای قبلی در حل MTSP باعث شد تا یک رویکرد مبتنی بر الگوریتم ژنتیک به نام الگوریتم ژنتیکی - مورچگان که به آن (AC-PGA) گفته می‌شود برای چندین نقطه‌ی شروع غیر ثابت ایجاد شود این رویکرد برای حل مسئله‌ی MTSP به نحوی در نظر گرفته شده است که فروشنده‌ها به جای یک نقطه‌ی شروع ثابت، از نقاط مختلف می‌توانند حرکت کنند و هر یک از آنها باید تور خود را در موقعیت شروع به کار خود پایان دهند.<sup>[۱۶]</sup>

الگوریتم پیشنهادی این مقاله، بهبود یافته‌ی الگوریتم AC-PGA است که نام آن IAC-PGA است. در این الگوریتم متغیرها به دو بخش تقسیم و به ترتیب حل می‌شوند. با این تفاوت که در الگوریتم هیبریدی پیشنهادی این مقاله، بعد از تعیین تعداد شهرهایی که هر فروشنده باید بازدید کند، متغیرهای قسمت اول (نقاط شروع هر فروشنده) از روی نقاط شکست بهترین کروموزوم الگوریتم ژنتیک ایجاد می‌شوند و به عنوان متغیرهای ورودی الگوریتم مورچگان در نظر گرفته می‌شوند. در الگوریتم مورچگان به منظور ایجاد تنوع و دست‌یابی به جواب‌های بهتر، از عملگر ۲-opt، برای جستجوی محلی با جابه‌جایی دوتایی روی تورهای ایجاد شده برای هر فروشنده (متغیرهای قسمت دوم) استفاده شده است. همچنین برای ایجاد تنوع بیشتر در الگوریتم ژنتیک، از عملگر جدید جهش بهره گرفته شده است. در الگوریتم مورچگان از یک روش متفاوت برای فرومون‌ریزی استفاده شده است. مقایسه‌ی IAC-PGA با سایر رویکردهای اکتشافی در مورد نمونه‌های معیار MTSP موجود در ادبیات نشان می‌دهد که رویکرد پیشنهادی از رویکردهای دیگر، به ویژه در مورد داده‌های در مقیاس بزرگ، بهتر است. نوآوری اصلی این مقاله تغییر در الگوریتم موجود است به نحوی که برای نمونه‌ی معیارهای مشخص در مقاله‌ی پایه<sup>[۱۶]</sup> کارایی بیشتری داشته باشد و علاوه بر آن به منظور نشان دادن کارا بودن الگوریتم جدید برای اولین بار برای ۱۰۰۰ شهر با تعداد فروشندگان مختلف مسئله اجرا شده و توانسته در زمان معقولی به یک جواب موجه دست پیدا کند.

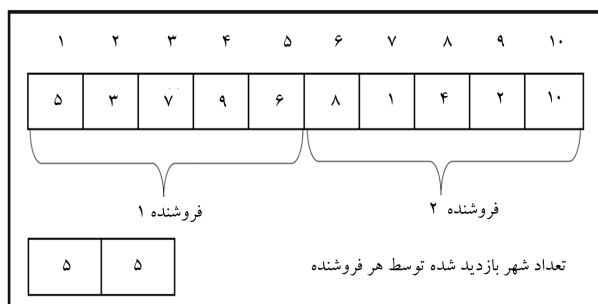
## ۳. تعریف مسئله

MTSP مورد نظر در این مقاله دارای چندین نقطه‌ی شروع غیر ثابت است و می‌توان به طور خلاصه به شرح زیر توصیف کرد:

یک گراف بدون جهت  $G=(V,A)$  در نظر بگیرید که  $V$  مجموعه گره‌ها و  $A$  مجموعه‌ی از قوس‌هاست که نمایانگر یال‌های موجود است. تابع هدف تقسیم  $V$  به زیرمجموعه‌هایی به تعداد فروشندگان است و هدف یافتن کمینه هزینه‌ی عبور از هر رأس در هر زیرمجموعه  $S_i$  است که رأس‌ها مربوط به فروشنده‌ی  $i$  است. تابع هدف MTSP را می‌توان چنین توصیف کرد:

$$\text{Min} \sum_{i=1}^m (d_{n^i,1}^i + \sum_{j=1}^{n^i-1} d_{j,j+1}^i) \quad (1)$$

عبارت اول نشان‌دهنده‌ی مسافت طی شده تمامی فروشندگان است و عبارت دوم مسافت طی شده فروشنده‌ی  $i$  است (شاخص اولین شهری که فروشنده‌ی  $i$  از آن بازدید کرده است  $1$  است و شاخص آخرین شهری که فروشنده‌ی  $i$  از آن بازدید کرده است  $n^i$  است) هر  $n^i$  باید کمینه و بیشینه تعداد شهرهایی را که هر فروشنده



شکل ۱. تعیین نقاط شکست روی بهترین کروموزوم.

۵	۳	۷	۹	۶
۸	۱	۴	۲	۱۰

شکل ۲. ماتریس مسیر بعد از تعیین نقاط شکست.

### ۲.۳. رویکرد IAC-PGA برای MTSP

در این بخش، یک الگوریتم ترکیبی جدید مبتنی بر PGA و ACO برای حل MTSP تهیه شده است. همان‌طور که قبلاً توضیح داده شد، MTSP متغیرهای زیادی دارد و از این رو الگوریتم طراحی شده آنها را به دو قسمت تقسیم می‌کند و از روش‌های مختلفی برای تعیین مقدار آنها استفاده می‌کند. PGA مورد بحث در این مقاله ممکن است به همگرایی زودرس برسد. با ترکیب مراحل الگوریتم‌های PGA و ACO، الگوریتم جدید بهبود یافته می‌تواند وابستگی ACO به مقادیر اولیه را کاهش دهد، از همگرایی زودرس جلوگیری کند و راه حل بهتری برای MTSP پیدا کند.

#### ۱.۲.۳. گام‌های الگوریتم PGA

از طریق الگوریتم PGA، بهترین کروموزوم که شامل یک توالی از همه شهرها که بر اساس کم‌ترین مسافت بین شهرهاست، استخراج می‌شود. سپس با تعیین تعداد شهری که هر فروشنده می‌تواند بازدید کند و ایجاد نقاط شکست روی بهترین کروموزوم شهرها، شروع متفاوت برای هر فروشنده مشخص می‌شود. در الگوریتم PGA هیچ عملگر تقاطعی وجود ندارد، عملگر تولید نسل بدین صورت است که در مرحله اول، ابتدا به نسبت  $\gamma$  از جمعیت اولیه جهت اعمال عملگر جهش انتخاب می‌شود و روی هر والد انتخاب شده، دو نقطه تصادفی انتخاب می‌شود و سپس توالی شهرها میان این دو نقطه معکوس می‌شود. این روش جهش از توالی شهر تکراری جلوگیری می‌کند و اشکالی در موجه بودن جواب ایجاد نمی‌شود.

کدگذاری راه حل: روش کدگذاری توالی، ساده‌ترین و کارآمدترین روش برای بیان راه حل MTSP است و از یک سری شماره شهر برای نشان دادن هر جواب استفاده می‌کند. برای کدگذاری توالی در MTSP از ماتریس مسیر و کروموزوم تعداد شهر بازدید شده برای هر فروشنده استفاده شده است.

به عنوان مثال، فرض کنید که دو فروشنده باید از ۱۰ شهر بازدید کنند و بهترین کروموزوم (دنباله مسیر) به صورت  $[5 \ 3 \ 7 \ 9 \ 6 \ 8 \ 1 \ 4 \ 2 \ 10]$  و دنباله‌ی تعداد شهر  $[5 \ 5]$  استخراجی از الگوریتم ژنتیک پارتو به صورت شکل ۱ باشد.

قسمت اول یک دنباله عدد صحیح از ۱ تا  $n$  است که مسیر را نشان می‌دهد و بخش دوم کروموزوم شامل  $m$  عدد است که هر عدد نشان‌دهنده‌ی تعداد شهر اختصاص یافته به هر فروشنده است. ماتریس توالی مسیر در شکل ۲ بعد از تعیین

باید از آن بازدید کند را برآورده کند، به طوری که مجموع شهرهای بازدید شده برابر با تعداد کل شهرها شود. نماد  $d_{j,j+1}^2$  فاصله‌ی بین شهر  $j$  و شهر بعدی را نشان می‌دهد که فروشنده‌ی  $i$  باید بازدید کند. [۱۱]

### ۱.۳. عملکرد الگوریتم ژنتیک پارتو و مورچگان

الگوریتم ژنتیک (GA) یک مدل محاسباتی است که نظریه‌ی تکامل بیولوژیکی داروین را شبیه‌سازی می‌کند و سپس بهترین راه حل را پیدا می‌کند. با توجه به عملگر تقاطع، در صورت ایجاد شهرهای تکراری که باعث نقض نیازهای مسئله می‌شوند، نمی‌توان از الگوریتم ژنتیک کلاسیک برای حل مستقیم TSP استفاده کرد. برخی از محققان بعضی از اپراتورهای تقاطع ویژه را توسعه داده‌اند که می‌توانند برای حل مسئله‌ی فروشنده‌ی دوره‌گرد، از طریق الگوریتم ژنتیک استفاده کنند. [۲۵] برای حل کمینه مسافت مسئله‌ی MTSP با چندین انبار دو الگوریتم ژنتیک پارتو (PGA) پیشنهاد شده است. یکی از آنها PGA با انتخاب رولت و انتخاب نخبه‌گراست که در آن چهار نوع جدید از عملیات جهش ارائه شده است. مورد دیگر به نام IPGA، انتخاب و جهش را به هم پیوند می‌دهد. یک اپراتور انتخاب جدید و یک اپراتور جهش جامع‌تر استفاده می‌شود.

ژن‌های هر فرد یک راه حل ممکن برای مسئله را نشان می‌دهد و تابع برازندگی هر فرد به معنای ارزش عملکرد در کروموزوم ارائه شده در PGA است. در فرایند انتخاب طبیعی، فرد با سازگاری بیشتر احتمال بالاتر برای زنده ماندن و تولید فرزندان دارد. این الگوریتم مشابه GA استاندارد است، به جز این که PGA، هر کودک را به جای دو نفر از والد فقط با یک فرد جداگانه تولید می‌کند. از این طریق می‌توان از تکرار در عملگر تقاطع در روند تولید نسل بعدی جلوگیری کرد.

در مورد مسئله‌ی تعریف شده، الگوریتم PGA برای حل مسئله استفاده می‌شود. با این حال، فرایند تولید مثل یک جستجوی محلی در یک منطقه‌ی کوچک اطراف کروموزوم‌هایی است که جمعیت فعلی از ژن فرزند (ناشی از جهش ژنتیکی والدین) وجود دارد. به عبارت دیگر، الگوریتم خیلی به جمعیت اولیه بستگی دارد و منطقه‌ی دیگر را در فضای حل نادیده می‌گیرد. با افزایش مقدار داده‌ها، فضای حل نیز ممکن است به سرعت افزایش یابد. در این حالت، اندازه‌ی ناحیه جستجو تغییر زیادی نخواهد کرد و اگر پارامترهای PGA تغییر نکنند، تعداد کروموزوم‌های نادیده گرفته شده تا حد زیادی افزایش می‌یابد. در نتیجه، کارایی الگوریتم بسیار پایین خواهد آمد.

الگوریتم کلونی مورچه‌ها [۳۰] با الهام از رفتار مورچه‌های واقعی در طبیعت ثابت کرده است که برای بسیاری از مسائل NP-hard قابل قبول است. به عنوان مثال، مسئله‌ی TSP توسط الگوریتم ACO به طور مؤثر حل شده است. [۳۲]

در مورد مسئله‌ی بهینه‌سازی، ACO با شبیه‌سازی فرایند فرومون‌ریزی مورچه‌ها، بهترین راه حل را پیدا می‌کند. راه حل امکان‌پذیر مسیر مورچه‌هاست و فضای حل مسئله را تشکیل می‌دهد. هم غلظت فرومون و هم طول مسیر می‌تواند در انتخاب سفر آن مورچه تأثیر بگذارد.

ویژگی‌های بیان شده توسط ACO نشان می‌دهد که برای حل مسئله‌ی TSP، الگوریتم ACO برای حل مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه با نقطه‌ی شروع ثابت مورد استفاده قرار گرفته است و نتایج آزمایش تأیید می‌کند که الگوریتم مورچگان بهتر می‌تواند چنین مسائلی را حل کند. [۳۱] با این حال، ACO توانایی تعیین نقطه‌ی شروع هر فروشنده را ندارد؛ بنابراین، نمی‌توان مستقیماً برای حل مسئله در نظر گرفته شده از آن استفاده کرد.

### ۲.۲.۳. گام‌های الگوریتم ACO

کروموزوم هر فرد را نمی‌توان مستقیماً به عنوان جواب برای MTSP بیان کرد، زیرا در فرایند ساخت جمعیت اولیه دو بردار تشکیل شده است. فرض کنید در MTSP تعداد فروشندگان و شهرهای مختلف وجود دارد، مراحل زیر برای یافتن جواب امکان‌پذیر برای هر فرد است:

۱. تعداد تکرار الگوریتم را تعیین کنید.

۲. فهرست ممنوعه به نام A و ماتریس فرمون  $\tau_{ij}$  را در نظر بگیرید. فهرست A شامل شهرهایی است که بازدید شده است. تمامی درایه‌های ماتریس فرمون در ابتدا برابر با یک است.

۳. مورچه روی نقطه‌ی شروع فروشنده قرار داده می‌شود.

۴. مورچه شروع به حرکت به شهر دیگر می‌کند و تصمیم می‌گیرد که بر اساس طول و غلظت فرمون هر مسیر ممکن، به کدام شهر برود. فرض کنید مورچه فعلی در شهر i است و احتمال حرکت به سمت شهر j،  $p_{ij}$  است. فرمول محاسبه عبارت است از:

$$p_{ij} = \begin{cases} \frac{\tau_{(i,j)}^\alpha c_{i,j}^\beta}{\sum_{k \notin A} \tau_{(i,k)}^\alpha c_{i,k}^\beta} & j \notin A \\ 0 & j \in A \end{cases} \quad (2)$$

که در آن  $\tau_{i,j}$  و  $C_{ij}$  به معنای غلظت فرمون و طول در مسیر بین شهر i و شهر j است و  $C_{ji}$  برابر است با معکوس مسافت از طول مسیر.  $\alpha$  و  $\beta$  به ترتیب اهمیت فرمون و طول مسیر را نشان می‌دهند. مورچه‌ی فعلی شهر بعدی را با استفاده از روش چرخ رولت و لیست ممنوعه A به روز می‌کند. پس از مشخص شدن مسیر هر مورچه (فروشنده) در هر تکرار (فروشنده) از روش جستجوی محلی 2-opt استفاده می‌شود و از بین آنها بهترین مسیر به منظور مقایسه‌ی نهایی انتخاب می‌شود. این روش بدین صورت است که به ازای هر تکرار تعدادی فروشنده وجود دارد که ابتدا فروشنده‌ی اول طبق نقطه‌ی شروع و تعداد شهر بازدید شده‌ی مشخص شروع به حرکت می‌کند و توالی شهرهای مورد نظر فروشنده طبق الگوریتم مورچگان ساخته می‌شود. سپس روش 2-opt بدین صورت عمل می‌کند که تمام جابه‌جایی‌های دوتایی بین شهرهای طی شده به صورتی که جای دو شهر و همچنین شهرهای میان آنها تغییر نکنند، انجام می‌شود و تابع هدف به‌ازای مسیر مورچه در هر بار جابه‌جایی تعیین می‌شود و بهترین توالی میان آنها با توجه به مقدار تابع هدف به عنوان بهترین تور آن فروشنده انتخاب می‌شود و سپس مورچه‌ی بعدی شروع به حرکت می‌کند و از همان روش برای انتخاب شهر بعدی استفاده می‌کند.

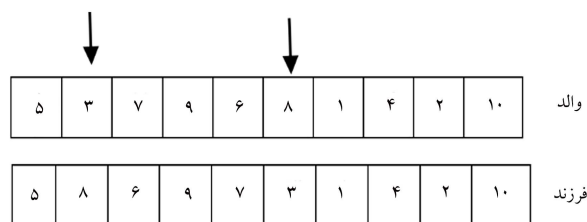
اگر مسیر همه مورچه‌ها (فروشنندگان) مشخص شود، ماتریس فرمون به‌روزرسانی می‌شود. قوانین به‌روزرسانی به شرح زیر است:

$$\tau(i, j) = (1 - \rho) * \tau(i, j) + \square \tau(i, j) \quad (3)$$

که در آن  $\rho$  نرخ تبخیر فرمون را نشان می‌دهد ( $0 < \rho < 1$ ):

$$\square \tau(i, j) = \frac{1}{D_{ij} + F_k^{ij}} \quad (4)$$

اگر K امین مورچه مسیری را بین شهرهای انتخاب شده باید بگذراند،  $F_k^{ij}$  برابر است با مجموع طول مسیر مورچه K و  $D_{ij}$  برابر با طول مسیر از شهر i به شهر j است.



شکل ۳. عملگر جهش.

نقاط شکست روی بهترین کروموزوم و مشخص شدن تعداد شهری که هر فروشنده باید بازدید کند، ایجاد می‌شود و شهر اول هر بردار (ژن اول) به عنوان نقاط شروع هر فروشنده در نظر گرفته می‌شود. این روش تضمین می‌کند که هر شهر فقط باید یک‌بار به جز شهر شروع توسط فروشنده بازدید شود.

تابع برازندگی: مقدار تابع برازندگی هر کروموزوم نشان‌دهنده‌ی سازگاری افراد با محیط است و سازگاری با ارزش همبستگی مثبت دارد. تابع برازندگی مورد استفاده در مقاله برگرفته از کل هزینه‌ی فاصله‌ی فروشندگان است. مسافت کوچک نشان‌گر راه‌حل خوب است و به احتمال زیاد حفظ می‌شود.

مقداردهی اولیه: به منظور مقداردهی اولیه در الگوریتم پیشنهادی مقاله ابتدا طبق الگوریتم PGA و تعداد تکرارهای مشخص مربوط به این الگوریتم و همچنین انجام عملگر جدید جهش که در شکل ۳ نشان داده شده است، بهترین تور (توالی کل مسیر بدون در نظر گرفتن چند فروشنده) انتخاب می‌شود. در مسئله‌ی MTSP متغیرها به دو قسمت تقسیم شده‌اند. متغیرهای قسمت اول نقاط شروع فروشندگان است در حالی که متغیرهای قسمت دوم توالی شهرهای بازدید شده برای هر فروشنده است. علاوه بر این، فرایند جمعیت اولیه فقط روی متغیرهای قسمت اول متمرکز است. در مرحله‌ی بعدی، تعداد شهرهایی که هر فروشنده باید بازدید کند، از قبل تعیین می‌شود؛ بنابراین، هر فرد دو کروموزوم کسب می‌کند که نقطه‌ی شروع هر فروشنده و تعداد شهرهایی است که فروشنده باید از آنها بازدید کنند. همچنین به اندازه‌ی جمعیت اولیه جفت آرایه‌های هم‌اندازه (تعداد خانه‌های هر آرایه به اندازه تعداد فروشندگان است) در نظر گرفته می‌شود که یکی از آنها تعداد شهرهای بازدید شده‌ی هر فروشنده و یکی دیگر نقاط شروع هر فروشنده است که این نقاط شروع پس از مشخص شدن تعداد شهر هر فروشنده، توسط بهترین تور تعیین می‌شود. سپس برای دنباله‌ی تعداد شهر و دنباله نقاط شروع، عملگر جهش برای ایجاد تنوع در شهرهای شروع و تعداد شهری که هر فروشنده باید بازدید کند بدین صورت که جای دو ژن در آرایه با یکدیگر عوض می‌شود، انجام می‌شود.

تعداد شهرهایی که هر فروشنده بازدید می‌کند باید بین  $n_1$  تا  $n_2$  مشخص شود، به صورتی که  $n_1$  و  $n_2$  حد پایین و بالایی از تعداد شهرهایی است که هر فروشنده به ترتیب باید بازدید کند. اعداد موجود در دنباله نقاط شروع باید با یکدیگر متفاوت باشند و هر مؤلفه در دنباله‌ی شماره‌ی شهر به طور تصادفی از ۱ تا n انتخاب می‌شود. برای پاسخگویی به الزامات مسئله، باید محدودیت‌های زیر رعایت شود: [۱۱]

۱. تعداد شهری که هر فروشنده باید بازدید کند باید از  $n_1$  بزرگ‌تر و از  $n_2$  کوچک‌تر باشد.

۲. مجموع تعداد شهرهای بازدید شده باید برابر با n باشد.

بعد از تعیین تعداد شهری که هر فروشنده باید بازدید کند و شهرهای شروع هر فروشنده در k تکرار، به گام‌های الگوریتم ACO بروید.

**Algorithm IAC-PGA.**  
**Input:** The number of city  $n$ , natural selection ratio  $\gamma$ , distance matrix  $D$ , pheromone volatilization rate  $\rho$ , the importance of pheromone and distance  $\alpha$  and  $\beta$ , ACO steps iteration  $k$ , maximum iteration  $N_0$   
 /PGA steps/  
 Create random cities sequences  
**For**  $i=1$  to  $N_0$   
 Calculate fitness value  
 Retain the individual with better fitness value on the ratio of  $\gamma$   
 Use the remaining ones with the new mutation operator  
**End for**  
 Select the best tour among the set of iterations  
**For**  $i=0$  to  $k$   
 Initialize  $N$  pairs of depot and city number sequence for each ant  
 Then select the starting point for each ant according to the best tour  
 mutation operator on the depot and city number sequence  
**end for**  
 /ACO steps/  
**For**  $i=0$  to  $k$   
 Initial pheromone matrix  $\tau$  and tabu list  $A$   
 Put an ant on each salesman's depot  
 Update tabu list  $A$   
 if current ant has finish its tour then  
 2-opt local search  
 Calculate the objective function of the path of an ant  
 select best tour for ant  
 Turn to the next ant  
 else  
 The next city is chosen according to the probability  $p_{ij}$   
**end if**  
**until**  $|A| = n$   
**end for**  
 Update pheromone matrix  $\tau$   
**until** The number of iterations reaches  $k$   
 Get a feasible solution  
**until** Perform ACO steps on each path  
**return** best solution found  
**Output:** Best solution found

شکل ۴. شبه کد الگوریتم IAC-PGA.

جدول ۱. نتایج آزمایش از طریق تغییر پارامترها.

$\rho$	$\alpha$	$\beta$	eil۵۱	eil۷۶	eil۱۰۱
۰/۱	۱	۲	۶۹۸	۹۳۴	۱۰۳۲
		۵	۴۹۰	۶۲۷	۸۱۴
		۸	۴۶۶	۶۱۷	۷۶۵
		۲	۷۰۲	۱۰۴۷	۱۴۲۷
	۵	۲	۴۹۳	۶۶۹	۸۱۱
		۵	۴۵۶	۵۹۷	۷۵۷
		۸	۴۷۸	۱۱۰۵	۱۲۹۸
		۲	۵۱۱	۶۷۵	۸۵۴
۰/۵	۱	۲	۶۷۸	۱۱۰۵	۱۲۹۸
		۵	۵۱۱	۶۷۵	۸۵۴
		۸	۴۷۶	۶۵۲	۷۶۳
		۲	۷۱۵	۱۰۵۱	۱۳۰۴
	۵	۲	۴۹۳	۶۸۹	۸۴۰
		۵	۴۹۸	۶۴۵	۷۶۳
		۸	۷۰۹	۹۹۷	۱۳۷۸
		۲	۵۰۹	۶۸۲	۸۶۵
۰/۷	۱	۲	۷۰۹	۹۹۷	۱۳۷۸
		۵	۵۰۹	۶۸۲	۸۶۵
		۸	۴۹۵	۶۴۲	۷۵۴
		۲	۷۱۰	۱۰۵۱	۱۳۴۵
	۵	۲	۵۲۳	۶۸۶	۸۵۴
		۵	۵۲۳	۶۸۶	۸۵۴
		۸	۴۹۷	۶۱۹	۸۴۹
		۲	۷۱۰	۱۰۵۱	۱۳۴۵

جدول ۲. تنظیم پارامتر.

پارامتر	مقدار
اندازه‌ی جمعیت	۱۰۰
IAC-PGA, AC-PGA تعداد تکرار	۱۰۰
$\rho$	۰/۱
$\alpha$	۲
$\beta$	۸
$\gamma(PGA)$	۰/۵

اگر تعداد تکرارهای الگوریتم تمام شود، مراحل الگوریتم ACO متوقف می‌شود و یک‌راه حل بهینه برای MTSP به دست می‌آید. شبه کد IAC-PGA در شکل ۴ آورده شده است.

## ۴. نتایج محاسباتی

در این بخش، تعداد زیادی از نمونه‌ها (در اندازه کوچک و اندازه بزرگ) برای مقایسات مسئله‌ی MTSP با الگوریتم‌های IWO، ABC<sup>[۱۵]</sup> و الگوریتم هیبریدی AC-PGA<sup>[۱۶]</sup> و همچنین الگوریتم هیبریدی بهبودیافته در این مقاله به نام IAC-PGA ارائه شده است. مسافت بین شهرها به صورت اقلیدسی محاسبه شده است. شبیه‌سازی این الگوریتم‌ها توسط MATLAB R۲۰۱۷a در یک رایانه‌ی شخصی با مشخصات پردازنده‌ی CPU ۲/۲۰ GHz Inter(R) Core(TM) i۳-۲۳۳۰ M و حافظه‌ی ۴GB کدگذاری شده است.

### ۱.۴. تنظیم پارامتر IAC-PGA برای MTSP

برای تنظیم پارامترهای ACO به منظور عملکرد بهتر، مقادیر مورد بررسی از منابع موجود<sup>[۳۰]</sup> اقتباس شده است. برای آزمایش شرایط مختلف الگوریتم باید اندازه پارامترها در چندین مقدار در نظر گرفته شود جدول ۱:

$\rho$ : حد وسط آن (۰/۵) و حد فوقانی (۰/۷) و حد پایینی (۰/۱) تنظیم شده است و این سه وضعیت این موضوع را نشان می‌دهد که باقی‌مانده‌ی فرومون به ترتیب با سرعت کند، متوسط و سریع تبخیر می‌شود.

$\alpha$ : در مقدارهای ۱ و ۲ تنظیم شده است، در حالی که  $\beta$  در مقادیر ۲، ۵ و ۸ تنظیم شده است.

از داده‌های eil۵۱، eil۷۶، eil۱۰۱ (دارای ۵۱، ۷۶ و ۱۰۱ شهر) در آزمون‌های مقایسه برای تنظیم پارامتر استفاده شده است. بعد از انجام برخی آزمایش‌های اولیه، اندازه جمعیت ۱۰۰ در نظر گرفته شده است. با ترکیب تنظیمات گفته شده در بالا، ۵۴ سناریو به دست می‌آید که برای عملکرد الگوریتم IAC-PGA مورد بررسی قرار گرفته است. برای هر سناریو، تعداد فروشنده‌گان برابر ۴ قرار داده شده است و سپس ۵ مرتبه الگوریتم اجرا می‌شود. همان‌طور که در جدول ۲ نشان داده شده است، نتایج با  $\beta = ۸$  نسبت به سناریوهای دیگر بهتر است. هنگامی که نمونه با تعداد شهر کم (eil۵۱) اجرا می‌شود، فاصله‌ی کمی بین نتایج وجود دارد؛ اما با افزایش اندازه داده‌ها، تفاوت بین نتایج به دست آمده در پارامترهای مختلف بیشتر می‌شود. بنابراین به نظر می‌رسد  $\alpha = ۲$  و  $\rho = ۰/۱$  بهترین ترکیب برای MTSP است و تنظیم پارامترها در الگوریتم‌ها، در جدول ۲ نشان داده شده است.

### ۲.۴. یک آزمایش مقایسه‌ی در الگوریتم‌های مختلف

در این بخش، الگوریتم‌های IAC-PGA و AC-PGA را برای مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه با ۱۰۰ شهر، ۵ فروشنده، کمینه و بیشینه تعداد شهرها که به ترتیب ۱۵ و ۳۰ هستند، آزمایش می‌شود. IAC-PGA در حال حاضر پس از تعداد معینی از تکرارها، به حالت همگرایی پایدار رسیده است (نمایش در شکل ۵). وابستگی بین کارایی الگوریتم پیشنهادی و انتخاب نقطه‌ی اولیه وجود ندارد و از این رو الگوریتم جدید توانایی جستجوی بالاتری برای حل MTSP دارد.

در اشکال ۶ و ۷، مسیرهای هر فروشنده با پارامترهای یکسان برای دو الگوریتم به

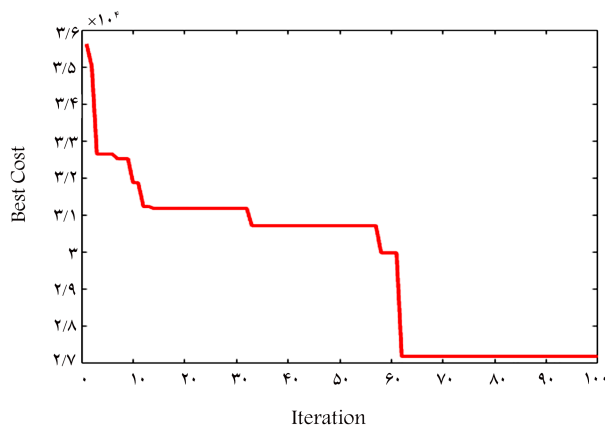
جدول ۳. نتایج در الگوریتم‌های متفاوت.

الگوریتم	مسافت کل (تابع هدف)	زمان حل (ثانیه)
PGA	۴۴۹۰۴	۲۸۲
IPGA	۳۷۸۱۷	۲۶۸
AC-PGA	۲۹۰۷۲	۲۵۲
IAC-PGA	۲۷۰۲۵	۶/۸۷

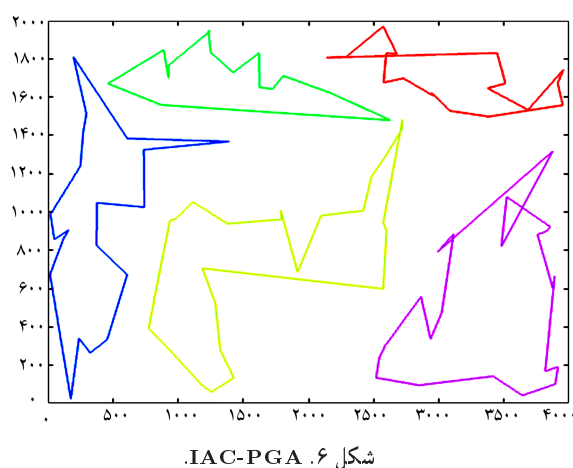
### ۳.۴. نتایج برای تعداد مختلف فروشندگان و تعداد شهرها

در مرحله‌ی بعدی، به بررسی تأثیر تعداد شهرها بر جواب پرداخته می‌شود و نتایج آزمایش سه الگوریتم با یکدیگر مقایسه می‌شوند. برای تجزیه و تحلیل نمونه‌های آزمایشی، الگوریتم IAC-PGA برای هر نمونه ۱۰ بار اجرا می‌شود. به منظور انجام آزمایش‌ها، تعداد تکرار و پارامترها در جدول ۳ آورده شده است. تعداد کمینه شهرهایی که هر فروشنده باید از آن بازدید کند برابر است با  $[N / (M + 1)]$  و بیشینه تعداد آن برابر  $[N / (M - 1)]$  است. با افزایش تعداد شهرها، پیچیدگی این مسئله به سرعت افزایش می‌یابد. از آنجا که در مقاله‌ی جیانگ<sup>[۱۶]</sup> شش الگوریتم موجود از لحاظ کیفیت جواب بررسی شده است، نتایج دو الگوریتم برتر با تعداد مختلف فروشندگان و شهرها در جدول ۴ گزارش شده است. نتایج برای الگوریتم‌های ABC و AC-PGA از مرجع<sup>[۱۶]</sup> به صورت مستقیم آورده شده است و الگوریتم پیشنهادی مقاله با همان داده‌ها و پارامترهای یکسان نمونه‌ها حل شده است. در جدول ۴، الگوریتم IAC-PGA در ۴ ستون که به ترتیب بهترین جواب، میانگین جواب، واریانس و متوسط زمان حل است، آورده شده است. نتایج نهایی از الگوریتم IAC-PGA با میانگین نتایج حاصل از دیگر الگوریتم‌ها مقایسه شده است. نتایج به دست آمده توسط الگوریتم‌های مختلف درمورد شهرهای کم‌تر بسیار نزدیک به یکدیگر است. با این حال، با افزایش تعداد شهرها، توانایی جستجوی سایر الگوریتم‌ها بسیار کاهش می‌یابد. در جدول ۴ میان سه الگوریتم در ۳۶ سناریوی موجود، جواب بهتر به صورت پررنگ نمایش داده شده است که در ۳۱ سناریو، الگوریتم پیشنهادی مقاله جواب بهتر را به دست می‌آورد و چون بین سه الگوریتم برترین نتایج حاصل شده است، بنابراین میان تمامی الگوریتم‌های موجود در ادبیات موضوع، برتری الگوریتم نوشتار حاضر نشان‌دهنده‌ی مناسب بودن الگوریتم IAC-PGA برای مسئله‌ی فروشنده‌ی دوره‌گرد چندگانه است. در شکل ۸ به عنوان مثال ۲ نمودار به منظور مقایسه از نمونه‌های متفاوت با اندازه تعداد شهر و تعداد فروشندگان مختلف ترسیم شده است و به ازای دیگر نمونه‌ها اعداد حل نمونه به صورت کامل در جدول ۴ آورده شده است.

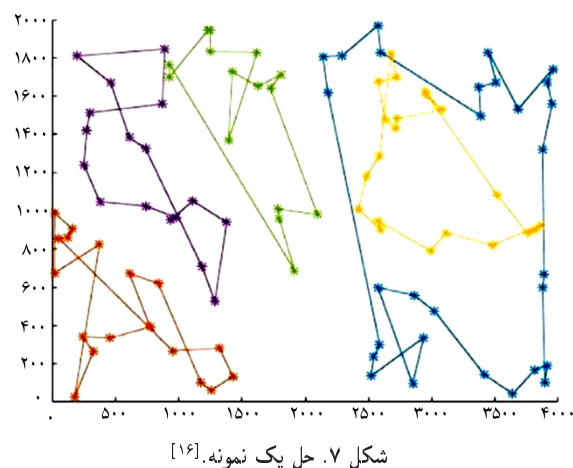
مشاهده می‌شود که الگوریتم پیشنهادی در هنگام بزرگ بودن تعداد شهرها از مزایای بسیار خوبی در یافتن جواب برخوردار است. به منظور مقایسه‌ی دو الگوریتم IAC-PGA، AC-PGA با تمام الگوریتم‌های موجود که در منبع<sup>[۱۶]</sup> آمده است، بهبود میان جواب هر کدام از آن دو الگوریتم برتر و بهترین جواب میان ۵ الگوریتم دیگر محاسبه شده است؛ در جدول ۵ میزان بهبود به صورت درصد آورده شده است. درصد‌های حساب شده بهبود میان بهترین میانگین پنج الگوریتم (۵ الگوریتم مقاله جیانگ به نام‌های IWO، IPGA، TWPS، PGA، ABC) است و میانگین الگوریتم IAC-PGA و AC-PGA است. به ازای هر تعداد شهر مختلف دو سطر داده وجود دارد که سطر اول مربوط به میزان بهبود الگوریتم IAC-PGA که اعداد مستقیم از مقاله‌ی جیانگ<sup>[۱۶]</sup> استخراج شده و سطر دوم بهبود مربوط به الگوریتم IAC-PGA است.



شکل ۵. همگرایی الگوریتم IAC-PGA.



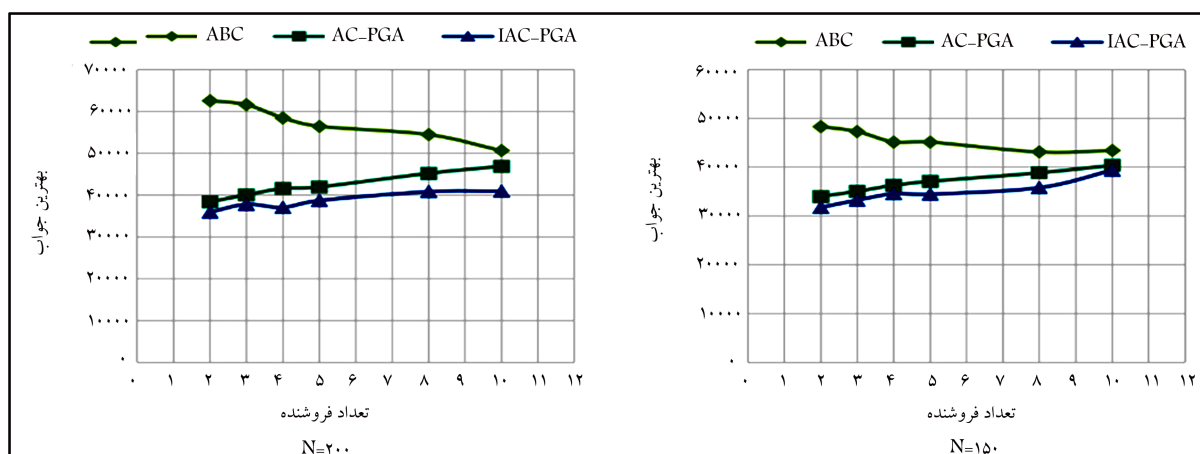
شکل ۶. IAC-PGA.

شکل ۷. حل یک نمونه<sup>[۱۶]</sup>.

عنوان نقشه مسیر ارائه شده است. نقشه‌ی مسیر الگوریتم AC-PGA در مرجع<sup>[۱۶]</sup> آورده شده است. نتایج الگوریتم‌ها در جدول ۳ نشان داده شده است. بدیهی است که الگوریتم جدید می‌تواند نتایج بهتری کسب کند و بدین معنی است که الگوریتم IAC-PGA نسبت به الگوریتم<sup>[۱۶]</sup> راندمان بالاتری برای MTSP دارد. نقشه مسیرهای فروشنده‌ها برای دو الگوریتم نشان می‌دهد که الگوریتم AC-PGA نسبت به الگوریتم IAC-PGA تداخل بیشتری در تورها دارد که در نهایت باعث ایجاد مسیر طولانی‌تری می‌شود و عدم تداخل تورهای IAC-PGA منجر به جواب‌های بهتر شده است.

جدول ۴. نتایج الگوریتم‌های متفاوت.

n	m	ABC			AC-PGA			IAC-PGA			متوسط زمان حل (ثانیه)
		بهترین جواب	میانگین	واریانس	بهترین جواب	میانگین	واریانس	بهترین جواب	میانگین	واریانس	
۵۱	۲	۴۸۲	۴۹۰	۱۰,۴	۴۵۷	۴۷۴	۱۱۰,۳	۴۵۶	۴۶۲	۹,۸	۳,۰۴
	۳	۴۷۵	۴۸۹	۹,۷	۴۶۷	۴۸۶	۹,۵	۴۸۱	۴۹۰	۸,۴	۲,۵۴
	۴	۴۷۷	۴۸۵	۱۱,۶	۴۷۳	۴۹۲	۱۱,۶	۵۰۷	۵۲۶	۹,۳	۲,۳۶
	۵	۴۷۴	۴۸۶	۱۱,۹	۴۸۸	۴۹۷	۱۰,۵	۵۱۲	۵۲۳	۸,۵	۲,۲۷
	۸	۴۷۰	۴۸۱	۱۲,۱	۵۲۱	۵۳۰	۱۱,۵	۵۱۱	۵۲۵	۸,۱	۲,۱۶
	۱۰	۴۷۲	۴۷۹	۱۱,۱	۵۳۲	۵۴۶	۸,۹	۴۹۷	۵۲۱	۹,۲	۱,۹۳
۱۰۰	۲	۳۰۸۱۶	۳۳۲۴۸	۱۰۶۰,۸	۲۶۹۰۲	۲۷۵۱۹	۴۹۳,۳	۲۳۵۸۵	۲۴۳۴۲	۴۷۵,۱	۹۰,۲
	۳	۳۰۱۳۷	۳۲۳۸۷	۹۴۰,۵	۲۶۴۱۶	۲۷۳۶۶	۵۷۴,۳	۲۵۷۶۵	۲۵۹۸۷	۳۱۶,۵	۷,۷۸
	۴	۳۰۵۳۹	۳۱۷۰۲	۷۵۳,۵	۲۷۳۲۲	۲۸۲۷۵	۶۲۲,۱	۲۶۸۸۱	۲۷۴۷۵	۴۵۳,۸	۷,۷۵
	۵	۲۹۷۷۲	۳۰۸۸۲	۷۳۲,۹	۲۸۷۵۹	۲۹۶۰۰	۶۲۲,۴	۲۷۰۲۵	۲۷۷۰۹	۵۳۲,۹	۶,۸۷
	۸	۲۸۶۲۶	۲۹۵۳۶	۹۴۴,۸	۳۱۳۳۲	۳۱۹۷۱	۶۱۷,۶	۲۸۵۷۹	۲۸۸۷۶	۳۴۲,۸	۶,۱۵
	۱۰	۲۸۹۶۶	۲۹۱۵۶	۹۲۱	۳۱۲۱۲	۳۲۲۹۳	۶۵۳,۷	۲۹۹۶۰	۳۰۹۷۶	۵۴۳,۷	۶,۰۱
۱۵۰	۲	۴۸۳۲۴	۵۰۷۰۹	۱۲۹۲,۹	۳۳۹۸۶	۳۴۹۲۳	۷۳۲,۴	۳۱۷۴۵	۳۱۹۸۷	۵۴۱,۲	۲۲,۳۶
	۳	۴۷۳۱۰	۴۹۵۸۵	۱۳۷۹,۳	۳۵۰۵۳	۳۶۰۴۱	۵۴۵,۲	۳۳۲۳۷	۳۴۵۴۳	۳۲۱,۴	۱۸,۹۲
	۴	۴۵۱۶۵	۴۸۴۷۵	۱۲۶۱,۳	۳۶۲۹۲	۳۷۶۸۷	۶۷۴,۹	۳۴۵۵۲	۳۵۰۱۲	۴۳۲,۵	۱۷,۱۰
	۵	۴۵۱۶۳	۴۶۹۵۸	۱۳۰۲,۶	۳۷۱۰۲	۳۷۶۷۹	۵۴۲,۹	۳۴۴۵۱	۳۵۹۸۷	۳۲۱,۵	۱۴,۷۱
	۸	۴۳۱۲۴	۴۴۳۸۱	۹۴۴,۷	۳۸۸۶۲	۴۰۹۴۲	۱۲۸۸,۲	۳۵۷۹۰	۳۶۲۱۴	۵۴۳,۲	۱۳,۰۷
	۱۰	۴۳۴۳۲	۴۵۴۸۶	۱۲۷۲,۷	۴۰۳۹۶	۴۲۰۳۶	۱۲۹۶,۶	۳۹۳۵۴	۴۰۲۳۸	۶۵۴,۱	۱۳,۹۶
۲۰۰	۲	۶۲۵۷۳	۶۴۹۲۰	۱۰۴۸,۷	۳۸۲۸۹	۳۹۷۸۸	۵۴۸,۷	۳۵۹۰۸	۳۶۶۱۷	۳۴۷,۵	۴۱,۵۴
	۳	۶۱۶۱۲	۶۳۲۵۷	۸۷۱,۹	۴۰۰۳۹	۴۱۴۱۶	۷۱۰,۷	۳۷۷۷۲	۳۸۳۱۳	۴۵۶,۴	۳۳,۳۱
	۴	۵۸۴۴۷	۶۰۶۴۲	۱۱۶۲,۳	۴۱۶۰۲	۴۲۴۲۲	۵۲۱,۲	۳۷۰۱۱	۳۸۴۲۵	۴۸۶,۵	۲۹,۴۱
	۵	۵۶۴۳۷	۵۸۳۵۸	۱۲۷۶,۸	۴۱۹۴۸	۴۲۹۳۴	۶۱۷,۲	۳۸۶۷۸	۳۹۲۳۴	۴۵۶,۷	۲۶,۸۹
	۸	۵۴۴۶۴	۵۶۰۸۸	۱۴۰۷,۵	۴۵۲۰۱	۴۶۸۹۶	۷۷۹,۶	۴۰۸۰۲	۴۱۲۱۳	۶۳۶,۵	۲۷,۴۲
	۱۰	۵۰۶۳۰	۵۳۵۱۹	۱۵۶۴,۴	۴۶۸۷۶	۴۷۷۲۱	۶۵۷,۹	۴۰۹۲۲	۴۱۸۳۵	۴۳۲,۶	۲۲,۶۷
۲۲۵	۲	۸۵۶۴	۹۱۱۲	۲۹۴,۱	۵۲۸۰	۵۳۷۳	۴۹,۲	۴۸۰۳	۴۸۷۶	۴۳,۲	۴۷,۸۵
	۳	۷۹۳۱	۸۴۰۹	۲۵۲,۸	۵۴۱۵	۵۵۷۴	۷۰,۸	۵۰۰۱	۵۰۱۳	۵۶,۴	۳۹,۶۴
	۴	۷۷۱۶	۷۹۸۲	۱۴۹,۱	۵۵۹۸	۵۶۶۳	۹۰,۹	۵۰۱۳	۵۰۸۷	۵۶,۷	۳۶,۲۶
	۵	۷۲۹۰	۷۶۷۷	۲۲۳,۷	۵۵۶۹	۵۷۴۱	۱۰۲,۹	۵۱۰۵	۵۱۲۴	۶۷,۸	۳۳,۸۲
	۸	۶۷۱۳	۷۱۶۳	۱۸۰,۶	۵۸۵۰	۶۰۰۲	۱۰۵,۵	۵۰۲۸	۵۱۹۸	۷۶,۵	۳۲,۵۲
	۱۰	۶۵۸۲	۶۹۴۹	۱۵۲,۷	۵۹۶۷	۶۱۸۴	۱۰۵,۵	۵۳۵۲	۵۴۷۵	۴۳,۲	۳۶,۱۴
۲۸۰	۲	۶۶۴۳	۶۹۳۸	۱۷۹,۲	۳۵۳۱	۳۶۲۴	۶۶,۹	۳۲۲۵	۳۴۳۲	۵۴,۳	۷۵,۴۴
	۳	۶۶۱۴	۶۶۵۹	۱۳۹,۳	۳۶۶۴	۳۷۵۳	۴۹,۹	۳۲۹۸	۳۴۵۳	۳۴,۵	۶۱,۸۰
	۴	۶۰۵۷	۶۳۱۰	۱۶۰,۸	۳۷۰۴	۳۸۲۸	۷۵,۶	۳۳۶۶	۳۴۵۳	۴۵,۶	۵۳,۴۹
	۵	۵۹۱۰	۶۱۱۲	۱۲۰,۸	۳۸۹۰	۳۹۸۳	۶۸,۱	۳۴۶۳	۳۵۷۶	۵۳,۴	۵۱,۲۳
	۸	۵۵۵۳	۵۶۶۹	۹۸,۴	۴۲۸۳	۴۳۸۱	۶۳,۴	۳۶۶۰	۳۷۲۱	۳۴,۶	۵۰,۲۱
	۱۰	۵۳۳۷	۵۴۹۳	۱۰۹,۶	۴۳۵۷	۴۴۳۵	۶۶,۱	۳۸۳۰	۳۷۲۳	۴۳,۲	۴۹,۸۱



شکل ۸. جواب‌های الگوریتم‌های متفاوت.

جدول ۵. بهبود الگوریتم.

الگوریتم	AC – PGA					
	IAC – PGA					
	۲	۳	۴	۵	۸	۱۰
۵۱	-۴/۹	-۰/۹۱	-۱/۴۲	-۲/۲۳	-۹/۱۹	-۱۲/۱۸
	-۲/۵	-۱/۸۳	-۴/۸	-۷/۷	-۷	-۸/۳
۱۰۰	۱۳/۴۹	۱۸/۳۵	۱۰/۸۱	۴/۱۵	-۷/۶۱	-۹/۸۴
	۱۹/۳	۲۴/۶۲	۱۵/۶۲	۱۵/۳	۲/۲۸	-۵/۸
۱۵۰	۳۱/۱۳	۲۷/۳	۲۲/۲۶	۱۹/۷۶	۵/۷۰	۳/۶۱
	۳۶/۹۰	۳۰/۰۳	۲۷/۷	۲۳/۳۶	۱۰/۱۱	۱۱/۵
۲۰۰	۳۸/۸۳	۳۴/۵۳	۳۰/۰۵	۲۶/۴۳	۱۶/۲۷	۱۰/۸۳
	۴۳/۵۰	۳۹/۵	۳۶/۶	۳۲/۷۷	۲۶/۵۲	۲۲/۶۷
۲۲۵	۴۱/۰۳	۳۳/۷۱	۲۹/۰۶	۲۵/۲۰	۱۶/۲۱	۱۱/۳۵
	۴۶/۴۸	۴۰/۳۸	۳۶/۲۶	۲۳/۲۵	۲۷/۴۳	۲۱/۲۱
۲۸۰	۴۷/۷۷	۴۳/۴۶	۳۹/۳۴	۳۴/۸۳	۲۲/۷۲	۱۹/۲۶
	۵۰/۵۳	۴۸/۱۴	۴۴/۵۴	۳۹/۲۴	۳۴/۳۶	۳۰/۲۷

مختلف به وجود آمده در نمونه‌های متفاوت، در ۳۳ مورد الگوریتم پیشنهادی دارای بهبود بیشتر بوده است.

آزمون غیر پارامتری فریدمن (Friedman) به منظور رتبه‌بندی الگوریتم‌ها و همچنین برای وجود تفاوت معناداری میان آنها با سطح اطمینان ۹۵٪ روی میانگین جواب‌های الگوریتم‌های متفاوت انجام شده است. این آزمون در نرم‌افزار SPSS ۱۶/۰۱ انجام شده است. چنان‌که در جدول ۷ نشان داده شده است اولین رتبه (۱/۵۶) مربوط به الگوریتم پیشنهادی مقاله است. همچنین در جدول ۸

همچنین به منظور مقایسه‌ی دو الگوریتم، نسبت الگوریتم AC-PGA به الگوریتم IAC-PGA و همچنین میزان بهبود الگوریتم پیشنهادی در جدول ۶ محاسبه شده است. چنان‌که مشاهده می‌شود در ۳۳ مورد این نسبت کم‌تر از ۱ شده است که نشان‌دهنده‌ی میزان تابع هدف بهتر برای الگوریتم IAC-PGA است. در ۳۳ سناریو دارای بهبود بوده است که گاهی به بیش از ۱۴ درصد نیز رسیده است. بین هر دو عدد به‌ازای هر تعداد شهر و تعداد فروشنده، عدد بزرگ‌تر (بهبود بیشتر) به صورت پررنگ نمایش داده شده است. چنان‌که مشاهده می‌شود از ۳۶ سناریوی



جدول ۶. مقایسه ی دو الگوریتم.

الگوریتم					
میزان بهبود	نسبت	IAC-PGA	AC-PGA	m	n
۰,۲	۰,۹۹۷	۴۵۶	۴۵۷	۲	۵۱
-۲	۱,۰۲۹	۴۸۱	۴۶۷	۳	
-۷,۱	۱,۰۷	۵۰۷	۴۷۳	۴	
-۴,۹	۱,۰۴۹	۵۱۲	۴۸۸	۵	
۱,۹۱	۰,۹۸	۵۱۱	۵۲۱	۸	
۶,۵۷	۰,۹۳	۴۹۷	۵۳۲	۱۰	
۱۲,۳۲	۰,۸۷	۲۳۵۸۵	۲۶۹۰۲	۲	۱۰۰
۲,۴۶	۰,۹۷	۲۵۷۶۵	۲۶۴۱۶	۳	
۱,۶۱	۰,۹۸	۲۶۸۸۱	۲۷۳۲۲	۴	
۶,۰۲	۰,۹۳	۲۷۰۲۵	۲۸۷۵۹	۵	
۸,۷۸	۰,۹۱	۲۸۵۷۹	۳۱۳۳۲	۸	
۴,۰۱	۰,۹۵	۲۹۹۶۰	۳۱۲۱۲	۱۰	
۶,۵	۰,۹۳	۳۱۷۴۵	۳۳۹۸۶	۲	۱۵۰
۵,۱	۰,۹۴	۳۳۳۳۷	۳۵۰۵۳	۳	
۴,۷۹	۰,۹۵	۳۴۵۵۲	۳۶۲۹۲	۴	
۷,۱	۰,۹۲	۳۴۴۵۱	۳۷۱۰۲	۵	
۷,۹	۰,۹۲	۳۵۷۹۰	۳۸۸۶۲	۸	
۲,۵	۰,۹۷	۳۹۳۵۴	۴۰۳۹۶	۱۰	
۶,۴	۰,۹۳	۳۵۹۰۸	۳۸۴۸۹	۲	۲۰۰
۵,۶	۰,۹۴	۳۷۷۷۲	۴۰۰۳۹	۳	
۱۱,۰۳	۰,۸۸	۳۷۰۱۱	۴۱۶۰۲	۴	
۷,۷	۰,۹۲	۳۸۶۷۸	۴۱۹۴۸	۵	
۹,۷	۰,۹۰	۴۰۸۰۲	۴۵۲۰۱	۸	
۱۲,۷	۰,۸۷	۴۰۹۲۲	۴۶۸۷۶	۱۰	
۹,۰۳	۰,۹۰	۴۸۰۳	۵۲۸۰	۲	۲۲۵
۷,۶	۰,۹۲	۵۰۰۱	۵۴۱۵	۳	
۱۰,۴	۰,۸۹	۵۰۱۳	۵۵۹۸	۴	
۸,۳	۰,۹۱	۵۱۰۵	۵۵۶۹	۵	
۱۴,۰۵	۰,۸۵	۵۰۲۸	۵۸۵۰	۸	
۱۰,۷	۰,۸۹	۵۳۵۲	۵۹۶۷	۱۰	
۸,۶	۰,۹۱	۳۲۲۵	۳۵۳۱	۲	۲۸۰
۱۰,۲	۰,۹	۳۲۹۸	۳۶۶۴	۳	
۹,۱	۰,۹	۳۳۶۶	۳۷۰۴	۴	
۱۰,۹	۰,۸۹	۳۴۶۳	۳۸۹۰	۵	
۱۴,۵	۰,۸۵	۳۶۶۰	۴۲۸۳	۸	
۱۲,۰۲	۰,۸۷	۳۸۳۰	۴۳۵۷	۱۰	

جدول ۷. رتبه بندی الگوریتم.

الگوریتم	Mean Rank
ABC	۲/۸۱
IAC-PGA	۱/۵۶
PGA	۵/۷۲
IPGA	۴/۲۸
TWPS	۷/۰۰
IWO	۴/۳۳
AC-PGA	۲/۳۱

جدول ۸. نتایج تحلیل آمار.

N	۳۶
Chi-Square	۱۷۳/۰۱۲
df	۶
Asymp. Sig	۰/۰۰۰

جدول ۹. نتایج ۱۰۰۰ شهر.

الگوریتم موجود در مقاله			
n	m	IAC – PGA (جواب)	زمان (ثانیه)
۱۰۰۰	۲	۲۵۳۶۷۵۹۵	۱۳۴۴/۷
	۳	۲۵۶۲۹۲۶۹	۱۱۳۲/۱۴
	۴	۲۵۳۰۱۷۰۷	۸۸۱/۹
	۵	۲۵۵۳۶۹۵۴	۸۳۳/۴
	۸	۲۶۹۲۱۷۰۷	۷۲۱/۹۸
	۱۰	۲۶۵۹۷۹۷۲	۷۱۰/۱۴

مقدار p-value زیر ۰/۰۵ است و این نشان دهنده ی تفاوت معنادار میان میانگین الگوریتم هاست.

همچنین برای اولین بار برای تعداد ۱۰۰۰ شهر و تعداد فروشندهگان مختلف این الگوریتم اجرا شده و نتایج آن در جدول ۹ آمده است. این نتایج نشان می دهد که باتوجه به بزرگ بودن ابعاد مسئله، الگوریتم توانسته است در زمان معقولی به جواب موجه برسد. از آن جا که در مقاله تا ۲۸۰ شهر بررسی شده است، تابع هدف موجود برای ۱۰۰۰ شهر بدون درصد بهبود آورده شده است.

## ۵. نتیجه گیری

در این مقاله، یک الگوریتم ترکیبی بهبود یافته به نام IAC-PGA با ترکیب ACO و PGA، برای حل مؤثر مسئله ی MTSP در مقیاس بزرگ طراحی شده است. دو نکته ی مهم وجود دارد: اول این که الگوریتم PGA وابستگی زیادی به مقدار جمعیت اولیه دارد و پس از ایجاد جمعیت اولیه، MTSP در مقیاس بزرگ را به طور مؤثر نمی تواند حل کند. دیگر این که ACO قادر به تعیین نقاط شروع فروشنده ها نیست. IAC-PGA یک الگوریتم دومرحله یی است. در مرحله ی اول، ابتدا با توجه به الگوریتم ژنتیک بهترین تور تشکیل شده با استفاده از معیار تابع برازندگی انتخاب و سپس به صورت تصادفی تعداد شهر بازدید شده توسط هر فروشنده تعیین می شود. سپس نقطه ی شروع هر فروشنده با استفاده از بهترین تور از طریق PGA تعیین

داشته است؛ بنابراین، از دیدگاه فاصله سفر، می‌توان نتیجه گرفت که IAC-PGA در مقایسه با برخی الگوریتم‌های شناخته شده‌ی فراابتکاری، عملکرد بهتری دارد. همچنین مقایسه‌ی میان کارایی دو الگوریتم AC-PGA و IAC-PGA صورت گرفته است و نتایج برتری الگوریتم پیشنهادی مقاله را نشان می‌دهد. به منظور پیشنهادات آتی می‌توان گفت عمده‌ی MTSP به صورت هدف واحد در نظر گرفته می‌شود، بنابراین می‌توان الگوریتم پیشنهادی را به صورت چندهدفه برای مسئله‌ی MTSP توسعه داد. در نظر گرفتن پهنج‌های زمانی برای هر شهر و محاسبه‌ی میزان زود کرد و دیر کرد یکی دیگر از مسائل جذاب است که در دنیای واقعی نیز بسیار کاربرد دارد. از آن‌جا که ممکن است فروشندگان هزینه‌های مختلفی داشته باشند، امکان انتخاب تعداد فروشندگان بر اساس کمینه‌سازی هزینه می‌تواند در زندگی روزمره اهمیت بسیاری داشته باشد.

می‌شود. در این مرحله از یک نوع جهش مؤثر برای تعداد شهرها برای هر فروشنده به منظور راه‌حل‌های بهتر ایجاد شده است. الگوریتم ACO در مرحله‌ی دوم برای شناسایی کوتاه‌ترین مسیر هر فروشنده و سپس به دست آوردن یک‌راه حل بهینه با استفاده از روش جستجوی محلی 2-opt برای مسئله استفاده می‌شود. در این‌جا هدف مقایسه‌ی عملکرد الگوریتم IAC-PGA با وجود کارآمدترین پارامتر و پنج الگوریتم فراابتکاری دیگر است که تاکنون در ادبیات موضوع وجود داشتند. تعداد مختلفی از شهرها و تعداد متفاوتی از فروشندگان در این تحلیل در نظر گرفته شده است. نتایج به دست آمده توسط الگوریتم IAC-PGA به طور قابل توجهی بهتر از نتایج به دست آمده توسط سایر الگوریتم‌ها در صورت وجود مسائل در مقیاس بزرگ است. از ۳۶ نمونه معیار موجود جواب به دست آمده توسط الگوریتم توسعه یافته IAC-PGA در مقایسه با الگوریتم‌های دیگر، ۳۲ نمونه معیار بهبود قابل توجهی

## پانوشته‌ها

1. traveling salesmen problem
2. multiple traveling salesman problem
3. improved ant colony-partheno genetic algorithm

## منابع (References)

1. Held, M., et al., "Aspects of the traveling salesman problem." *IBM journal of Research and Development*, **28**(4), pp. 476-486 (1984).
2. Gouveia, L., Leitner, M. and Ruthmair, M. "Extended formulations and branch-and-cut algorithms for the black-and-white traveling salesman problem." *European Journal of Operational Research*, **262**(3), pp. 908-928 (2017).
3. Asadpour, A., et al. "An  $O(\log n / \log \log n)$ -approximation algorithm for the asymmetric traveling salesman problem." *Operations Research*, **65**(4), pp. 1043-1061 (2017).
4. Maity, S., Roy, A. and Maiti, M. "An imprecise multi-objective genetic algorithm for uncertain constrained multi-objective solid travelling salesman problem." *Expert Systems With Applications*, **46**, pp.196-223 (2016).
5. Nagata, Y. and Soler, D. "A new genetic algorithm for the asymmetric traveling salesman problem." *Expert Systems with Applications*, **39**(10), pp. 8947-8953 (2012).
6. Choong, S.S., Wong, L.-P. and Lim, C.P. "An artificial bee colony algorithm with a modified choice function for the traveling salesman problem." *Swarm and Evolutionary Computation*, **44**, pp. 622-635 (2019).
7. Khan, I. and Maiti, M.K. "A swap sequence based artificial bee colony algorithm for traveling salesman problem." *Swarm and Evolutionary Computation*, **44**, pp. 428-438(2019).
8. Chen, S.-M. and Chien, C.-Y. "Parallelized genetic ant colony systems for solving the traveling salesman problem." *Expert Systems with Applications*, **38**(4), pp. 3873-3883 (2011).
9. Escario, J.B., Jimenez, J.F. and Giron-Sierra, J.M. "Ant colony extended: experiments on the travelling salesman problem." *Expert Systems with Applications*, **42**(1), pp. 390-410 (2015).
10. Osaba, E., et al. "A discrete water cycle algorithm for solving the symmetric and asymmetric traveling salesman problem." *Applied Soft Computing*, **71**, pp. 277-290 (2018).
11. Zhong, Y., et al. "Discrete comprehensive learning particle swarm optimization algorithm with metropolis acceptance criterion for traveling salesman problem." *Swarm and Evolutionary Computation*, **42**, pp. 77-88 (2018).
12. Ezugwu, A.E.-S., Adewumi, A.O. and Frincu, M.E. "Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem." *Expert Systems with Applications*, **77**, pp. 189-210 (2017).
13. Yuan, S., et al. "A new crossover approach for solving the multiple travelling salesman problem using genetic algorithms." *European Journal of Operational Research*, **228**(1), pp. 72-82 (2013).
14. Chen, Y., et al. "A modified two-part wolf pack search algorithm for the multiple traveling salesmen problem." *Applied Soft Computing*, **61**, pp. 714-725 (2017).
15. Venkatesh, P. and Singh, A. "Two metaheuristic approaches for the multiple traveling salesperson problem." *Applied Soft Computing*, **26**, pp. 74-89 (2015).
16. Jiang, C., Wan, Z. and Peng, Z. "A new efficient hybrid algorithm for large scale multiple traveling salesman problems." *Expert Systems with Applications*, **139**, 112867 (2020).
17. Tang, L., et al. "A multiple traveling salesman problem model for hot rolling scheduling in shanghai baoshan iron

- & steel complex." *European Journal of Operational Research*, **124**(2), pp. 267-282 (2000).
18. Kota, L. and Jarmai, K. "Mathematical modeling of multiple tour multiple traveling salesman problem using evolutionary programming." *Applied Mathematical Modelling*, **39**(12), pp. 3410-3433 (2015).
19. Shiri, S. and Huynh, N. "Optimization of drayage operations with time-window constraints." *International Journal of Production Economics*, **176**, pp. 7-20 (2016).
20. Király, A., et al. "Minimization of off-grade production in multi-site multi-product plants by solving multiple traveling salesman problem." *Journal of cleaner production*, **111**, pp. 253-261 (2016).
21. Gavish, B. and Srikanth, K. "An optimal solution method for large-scale multiple traveling salesmen problems". *Operations Research*, **34**(5), pp. 698-717 (1986).
22. Bektas, T. "The multiple traveling salesman problem: an overview of formulations and solution procedures." *Omega*, **34**(3), pp. 209-219 (2006).
23. Oberlin, P., Rathinam, S. and Darbha, S. "A transformation for a multiple depot, multiple traveling salesman problem." In *2009 American Control Conference*. IEEE (2009).
24. Pandiri, V. and Singh, A. "A hyper-heuristic based artificial bee colony algorithm for k-interconnected multi-depot multi-traveling salesman problem." *Information Sciences*, **463**, pp. 261-281 (2018).
25. Zhou, H. Song, M. and Pedrycz, W. "A comparative study of improved GA and PSO in solving multiple traveling salesmen problem." *Applied Soft Computing*, **64**, pp. 564-580 (2018).
26. Zhu, Y. and Wu, L. "Structure study of multiple traveling salesman problem using genetic algorithm." In *2019 34th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE (2019).
27. Mitchell, M. "An Introduction to genetic algorithms", *Massachusetts Institute of Technology*, MIT Press, Cambridge (1996).
28. Alves, R.M. and Lopes. C.R. "Using genetic algorithms to minimize the distance and balance the routes for the multiple traveling salesman problem." In *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE (2015).
29. Lo, K.-M., et al. "A genetic algorithm with new local operators for multiple traveling salesman problems." *International Journal of Computational Intelligence Systems*, **11**(1), pp. 692-705 (2018).
30. Dorigo, M., Colorni, A. and Maniezzo, V. "Distributed optimization by ant colonies" (1991).
31. Ghafurian, S. and Javadian, N. "An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems." *Applied Soft Computing*, **11**(1), pp. 1256-1262 (2011).
32. Changdar, C., Pal, R.K. and Mahapatra, G. "A genetic ant colony optimization based algorithm for solid multiple travelling salesmen problem in fuzzy rough environment." *Soft Computing*, **21**(16), pp. 4661-4675 (2017).
33. Dorigo, M., Maniezzo, V. and Colorni, A. "Ant system: optimization by a colony of cooperating agents." *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **26**(1), pp. 29-41 (1996).