مهندسی صنایع و مدیریت شریف، (تابستان ۱۳۹۳)
دوره‌ی ۱-۳۰، شماره‌ی ۱/۱، ص. ۱۲۱-۱۳۰، (یادداشت فنی)

Research Note

حل مسئله‌ی چندین فروشنده‌ی دوره‌گرد به وسیله‌ی الگوریتم ICA

مجید یوسفی‌خوشبخت * **(کارشناس ارشد)**
دانشکده ریاضی، دانشگاه آزاد اسلامی، واحد همدان
سیدناصر هاشمی (استادیار)
دانشکده‌ی ریاضی و علوم کامپیوتر، دانشگاه صنعتی امیرکبیر

مسئله‌ی چندین فروشنده‌ی دوره‌گرد (MTSP)[1] گسترشی است مشهور از مسئله‌ی فروشنده‌ی دوره‌گرد (TSP)[2]، که ثابت شده یک مسئله‌ی NP-hard است. اگرچه MTSP یک مسئله‌ی پیچیده‌ی بهینه‌سازی ترکیباتی است، می‌توان آن را به مسائل گوناگونی در مسیریابی و زمان‌بندی توسعه داد. به‌علاوه، تحقیقات روی این مسئله، برخلاف TSP که گستردگی آن توجه بسیار زیادی را به خود معطوف کرده، بسیار محدود بوده است. از این‌رو، در این نوشتار یک روش جدید بهینه‌سازی به‌نام «الگوریتم رقابت بهره‌جویانه (ICA)» برای حل این مسئله ارائه می‌شود. این روش ملهم از رقابت کشورهای مستقل و وابسته برای حل مسائل بهینه‌سازی ترکیباتی است. الگوریتم پیشنهادی روی دو دسته مسائل از ادبیات موضوع مورد آزمایش قرار گرفته است. نتایج محاسباتی نشان می‌دهد که الگوریتم رقابت خوبی با دیگر الگوریتم‌های فرابتکاری برای حل مسائل MTSP داشته است. همچنین چندین جواب بهینه به‌وسیله‌ی الگوریتم پیشنهادی به دست آمد.

واژگان کلیدی: مسئله‌ی چندین فروشنده‌ی دوره‌گرد، الگوریتم رقابت بهره‌جویانه، مسائل بهینه‌سازی ترکیباتی، مسئله‌ی مسیریابی وسیله‌ی نقلیه.

khoshbakht@iauh.ac.ir
nhashemi@aut.ac.ir

۱. مقدمه

مسئله‌ی چندین فروشنده‌ی دوره‌گرد (MTSP) همان مسئله‌ی مسیریابی وسیله‌ی نقلیه (VRP)[3] است که در آن ظرفیت وسایل نقلیه نامتناهی فرض شده است. بنابراین می‌توان به‌جای وسایل نقلیه از فروشنده‌های دوره‌گرد استفاده کرد. در این مسئله $m < n$ فروشنده از گره‌یی مشخص به‌نام انبار شروع به حرکت می‌کنند و بعد از ملاقات n گره به محل حرکت خود بازمی‌گردند؛ به شرط آن که هر گره فقط یک‌بار به‌وسیله‌ی یکی از فروشندگان ملاقات شود. هدف این مسئله تعیین مجموعه‌یی از مسیرهاست به‌طوری که هزینه‌ی کلی آن‌ها کمینه شود. در مسئله‌ی MTSP باید علاوه بر این که ترتیب گره‌ها برای هر فروشنده معین می‌شود، تعداد و گره‌های اختصاص‌یافته به هر فروشنده نیز تعیین شود. بنابراین چون مسئله‌ی فروشنده‌ی دوره‌گرد (TSP) یک مسئله‌ی NP-hard است، مسئله‌ی MTSP نیز به این دسته از مسائل تعلق دارد.

به‌علاوه مسئله‌ی MTSP به‌عنوان یکی از مهم‌ترین گسترش‌های مسئله‌ی TSP، از جایگاه بسیار مهمی در مسائل بهینه‌سازی ترکیباتی برخوردار است و بسیاری از مسائل با اضافه‌کردن محدودیت‌های جدید بر این مسئله ایجاد شده‌اند. عمده‌ی مواردی که به پیدایش نسخه‌های دیگر از مسئله‌ی MTSP می‌انجامد عبارت‌اند از:

تک‌انباره‌ها یا انباره‌های چندگانه: در MTSP تک‌انباره (SDMTSP)[4]، تمامی فروشندگان از یک انبار شروع به حرکت می‌کنند و بعد از ملاقات‌کردن سایر گره‌ها به همان انبار بازمی‌گردند. این در حالی است که در MTSP چندانباره (MDMTSP)[5]، فروشنده‌ها در انبارهای متفاوت قرار دارند. حال اگر این فروشنده‌ها بعد از انجام کار خود به همان انبار بازگردند، آن را MTSP چندانباره‌ی ثابت می‌نامند. اگر همین فروشنده‌ها بعد از ملاقات گره‌ها به انبارهای متفاوت بروند، به شرط آن که تعداد فروشنده‌های شروع کننده در هر انبار بعد از اتمام کار نیز ثابت بماند، آن را MTSP چندانباره‌ی غیرثابت می‌نامند.

تعداد فروشنده‌ها: تعداد فروشنده‌های هر مسئله در MTSP را در ابتدا ثابت یا به‌صورت یک متغیر کران‌دار در نظر می‌گیرند.

هزینه‌ی ثابت: هنگامی که تعداد فروشنده‌ها را در مسئله ثابت در نظر نگیرند، هر فروشنده‌ی مورد استفاده یک هزینه‌ی تحمیل شده ایجاد می‌کند. در این مورد کمینه‌کردن تعداد فروشنده‌ها نیز در تابع هدف مد نظر قرار می‌گیرد تا جواب بهتری برای مسئله به دست آید.

پنجره‌های زمانی: در این نسخه‌ی MTSP هر گره باید در یک بازه زمانی مورد ملاقات قرار گیرد که به آن مسئله‌ی چندین فروشنده‌ی دوره‌گرد با پنجره زمانی

* نویسنده مسئول
تاریخ: دریافت ۱۳۹۰/۱۱/۳، اصلاحیه ۱۳۹۱/۲/۱۷، پذیرش ۱۳۹۱/۳/۲.

(MTSPTW)[6] گفته می‌شود. این نسخه یکی از مهم‌ترین گسترش‌های MTSP است که کاربردهای فراوانی دارد. نمونه‌یی از کاربردهای این مسئله، مسیریابی اتوبوس مدارس و زمان‌بندی هواپیماها و کشتی‌هاست.

دیگر محدودیت‌های خاص: این محدودیت‌ها شامل بیشینه گره‌هایی است که هر فروشنده می‌تواند ملاقات کند، حداکثر یا حداقل مسافتی است که هر فروشنده باید طی کند.

امروزه به‌علت گستردگی MTSP و کاربردهای فراوان آن در بسیاری از مسائل بهینه‌سازی ترکیباتی ــ مانند زمان‌بندی وسیله‌ی نقلیه[7]،[1] مسیریابی وسیله‌ی نقلیه و دیگر مسائل مربوط به آن ــ مانند دریافت و تحویل همزمان کالا،[2] پنجره‌های زمانی،[3] سیستم امنیتی شبانه[8][4] و زمان‌بندی ماشین چاپ تبلیغات روزنامه[9][5] ــ مورد توجه قرار گرفته است. نمونه‌یی از کاربردهای مهم این مسئله در ادامه تشریح می‌شود.

زمان‌بندی چاپ مطبوعات: از نخستین کاربردهای MTSP می‌توان به زمان‌بندی چاپ مطبوعات دوره‌یی به‌وسیله‌ی چندین ناشر[6] اشاره کرد که طی آن پنج جفت سیلندر بین رول‌های کاغذ قرار می‌گرفت و بدین‌ترتیب چاپ به‌طور همزمان بر دو روی کاغذ انجام می‌گرفت. این ماشین‌های چاپ در سه نوع ۴، ۶ و ۸ صفحه‌یی مورد استفاده قرار می‌گرفت. مسئله‌ی زمان‌بندی شامل تصمیم‌گیری برای این ماشین‌هاست که چه موقع روشن شوند و چه مدت کار کنند.

زمان‌بندی خدمه: یکی دیگر از موارد کاربرد MTSP در جابه‌جایی پس‌اندازها بین شعب بانک گزارش شده است.[7] در این مورد، پس‌اندازها باید پس از جمع‌آوری از شعب بانک توسط خدمه به شعبه مرکزی منتقل شود. هدف این مسئله کمینه‌کردن مسیرهای پیموده شده به‌وسیله‌ی خدمه‌هاست.

مسئله‌ی مسیریابی اتوبوس مدارس: این مسئله یکی از نسخه‌های MTSP است که با محدودیت‌هایی همراه است.[8] هدف، یافتن گونه‌یی زمان‌بندی برای بارگذاری اتوبوس‌هاست که سبب کمینه‌شدن تعداد مسیرها، کمینه‌شدن مسیرهای پیموده‌شده به‌وسیله‌ی اتوبوس‌ها، و عدم تجاوز از ظرفیت اتوبوس‌ها می‌شود. به‌علاوه در این مسئله نباید زمان لازم برای پیمایش هر مسیر از یک مقدار معین بیشتر شود.

زمان‌بندی مصاحبه: در کاربرد دیگری که برای MTSP مطرح شد،[9] این مسئله برای زمان‌بندی مصاحبه بین دلال‌ها و فروشنده‌ها در صنعت توریسم مورد استفاده قرار گرفت. در این مسئله هر دلال نقش فروشنده‌یی را بازی می‌کند که باید تعدادی از غرفه‌های فروشندگان دیگر را ملاقات کند؛ این امر به‌صورت مجموعه‌یی از T شهر نشان داده شده است.

طراحی مأموریت: این مسئله در ابتدا در مفهوم ربات‌های مستقل موبایل ارائه شد که کاربردهایی چون کشف انبارهای مستقل جنگی، مراکز پستی مستقل و انفجارهای نجومی دارد. طراحی مأموریت شامل معین‌کردن مسیرهای بهینه برای هر ربات است به‌گونه‌یی که اهداف مأموریت را در کوتاه‌ترین زمان ممکن انجام دهد. باید توجه کرد که طراحی مأموریت نسخه‌یی از MTSP را مورد استفاده قرار می‌دهد که در آن n ربات باید m هدف را ملاقات کند و در انتها به محل حرکت خود بازگردد. محققین کاربرد MTSP را علاوه بر طراحی مأموریت،[10] در محیط‌های نامنظم[11] نیز ارائه کردند.

زمان‌بندی غلتک‌های داغ: در صنعت استیل و آهن انجام سفارشات بر غلتک‌های داغ و برای یک شیفت تولید باید زمان‌بندی شود، به‌طوری که هزینه‌ی کلی راه‌اندازی مسیر تولید کمینه شود. یکی از کاربردهای جدید MTSP در کشور چین و روی مسئله‌ی زمان‌بندی غلتک‌های داغ گزارش شد.[12]

طراحی شبکه‌های بررسی سیستم‌های ماهواره‌یی سراسری هوانوردی: در سال‌های اخیر کاربرد بسیار جالبی برای MTSP [13] ارائه شد که در آن از این مسئله در سیستم‌های ماهواره‌یی هوانوردی استفاده می‌شود. سیستم ماهواره‌یی سراسری هوانوردی سیستمی است که در آن برای شناسایی تمامی مکان‌های جهانی یک همگرایی فراهم می‌شود. این مسئله، به‌خصوص درمورد حادثه‌هایی بزرگ همچون سیل و زلزله، بسیار کاربردی محسوب می‌شود. یادآور می‌شود که کاربرد این مسئله به موارد ذکر شده محدود نمی‌شود و در موارد متعدد دیگری می‌توان از آن بهره برد.[14]

در دو دهه‌ی گذشته مسئله‌ی فروشنده‌ی دوره‌گرد توجه بسیاری را به خود جلب کرده، و بسیاری از روش‌ها مانند روش‌های شاخه و کران،[15] صفحات برش،[16] شبکه‌های عصبی[17] و جست‌وجوی ممنوع[18] برای حل این مسئله پیشنهاد شده‌اند. این روش‌ها، روش‌هایی دقیق و تخمینی‌اند که در زمانی طولانی به جواب بهینه می‌رسند، یا به یک جواب زیربهینه در یک زمان قابل قبول دست می‌یابند. مسئله‌ی چندین فروشنده‌ی دوره‌گرد (MTSP) برخلاف مسئله‌ی فروشنده‌ی دوره‌گرد (TSP) کم‌تر مورد توجه قرار گرفته است؛ با این حال مانند TSP، از الگوریتم‌های دقیق و تخمینی برای حل آن‌ها استفاده می‌شود. در الگوریتم‌های دقیق مانند روش تخفیف لاگرانژی یدپالی[10] و همکارانش،[19] روش شبه تخصیص گرومیکو[11] و همکارانش[20] که بیشتر برای مسائل با اندازه‌ی تقریباً کوچک به کار می‌روند، جواب بهینه‌ی مسئله به دست می‌آید علی‌رغم آنکه زمان زیادی برای محاسبه‌ی جواب باید صرف شود. روش‌های دقیق دیگری نیز وجود دارد که ایده‌ی اصلی آن‌ها تخفیف چندین محدودیت و ساده‌سازی مسئله‌ی مربوطه است،[21] مانند روش‌های مبتنی بر شاخه و کران.[22،23] در این خصوص، در مطالعه‌یی دیگر کاربرد روش شاخه و کران در مسئله‌ی MTSP بررسی شد.[24]

از آنجا که MTSP یک مسئله‌ی پیچیده‌ی بهینه‌سازی از نوع NP-hard است، الگوریتم‌های دقیق کارایی مناسبی در کاربردهای واقعی و در اندازه مورد نیاز در دنیای امروزی ندارند و نمی‌توانند در یک زمان قابل قبول به جواب‌های بهینه دست یابند. درنتیجه برای حل این‌گونه از مسائل از الگوریتم‌های ابتکاری و فراابتکاری استفاده می‌شود که می‌توانند جواب زیربهینه را در یک زمان قابل قبول به دست آورند. بعضی از الگوریتم‌های ابتکاری که در زمان مناسب به جواب زیربهینه می‌رسند، فاقد ساختار لازم برای فرار از نقاط بهینه‌ی محلی‌اند و معمولاً به جواب‌های خوبی همگرا نمی‌شوند. نمونه‌ی این الگوریتم‌ها، اولین الگوریتم ارائه شده برای حل مسئله‌ی چندین فروشنده‌ی دوره‌گرد (MTSP)[25] و نیز یکی از قدیمی‌ترین روش‌های ابتکاری ارائه‌شده توسط راسل[12] برای حل MTSP است.[26] برای حل این مسئله روش‌های دیگری نیز ارائه شده است.[27] از طرف دیگر در سال‌های اخیر الگوریتم‌هایی پیشنهاد شده که دارای ساختار تصادفی در تعیین جواب هستند. این الگوریتم‌ها که «فراابتکاری» نامیده می‌شوند می‌توانند تا حد امکان از بهینه‌های محلی دور و به جواب‌های بسیار خوب همگرا شوند. به‌طور مثال می‌توان به روش‌های شبکه‌های عصبی،[28،29] الگوریتم ژنتیک[30] و نیز روش جست‌وجوی ممنوع[31] اشاره کرد. در سال‌های اخیر روش الگوریتم ژنتیک به‌طور موفقیت‌آمیز برای حل مسئله‌ی TSP مورد استفاده قرار گرفته[32] و روش‌های ژنتیک بر روی مسئله‌ی TSP مورد بررسی قرار گرفته است. این روش برای مسئله‌ی MTSP نیز مورد استفاده قرار گرفت.[33] بیشتر کارهای انجام شده روی مسئله‌ی MTSP

با استفاده از الگوریتم ژنتیک و برای کاربرد در مسئله‌ی زمان‌بندی وسیله‌ی نقلیه (VSP) بوده است.[۳۴،۳۵] این مسئله به‌طور معمول محدودیت‌هایی را شامل می‌شود، نظیر ظرفیت وسیله‌ی نقلیه، تعیین بیشینه شهرهایی که یک وسیله می‌تواند آن را ملاقات کند، محدودیت پنجره زمانی که در آن هر وسیله فقط در یک بازه زمانی خاص بارگیری می‌شود.در کاربردهای دیگر روش الگوریتم ژنتیک برای زمان‌بندی غلتک‌های داغ[۱۳] مورد استفاده قرار گرفته است.[۳۶] بدین‌منظور، ابتدا این مسئله به MTSP و سپس به TSP تبدیل، و به‌وسیله‌ی الگوریتم مورد نظر حل شده است. از روش الگوریتم ژنتیک برای حل MTSP در طراحی مسیر نیز استفاده شده است.[۳۷]

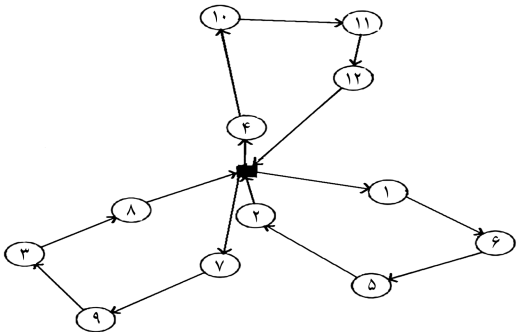
چنان که بیان شد، MTSP را می‌توان براساس جواب به این سؤال که «آیا فروشنده‌ها از یک انبار یا بیشتر شروع به حرکت می‌کنند؟» گسترش داد. چندین تحول در مسئله‌ی چندین فروشنده‌ی دوره‌گرد با یک انبار (SDMTSP) در ادبیات موضوع وجود دارد. به‌طور مثال در مطالعه‌یی پیرامون SDMTSP هر فروشنده سرویس‌دهی را با هزینه‌ی اختصاصی و متفاوت انجام می‌داد[۳۸] و هزینه‌ی یال‌ها لازم نبود که در شرایط نامساوی مثلثی صادق باشد. از آنجا که هدفِ این مسئله کاهش مجموع هزینه‌ی انجام‌شده به‌وسیله‌ی فروشندگان است، ممکن است تمامی فروشندگان در این مسئله مورد استفاده قرار نگیرند؛ برای حل این مسئله، SDMTSP به یک TSP نامتقارن تبدیل شد.[۳۸] از طرف دیگر، تبدیل کارایی برای همین مسئله انجام شد[۳۹] و با در نظر گرفتن نسخه‌ی متقارن SDMTSP مورد بررسی قرار گرفت.[۴۰] این در حالی است که یک تبدیل بهبودیافته برای SDMTSP متقارن ارائه شد که در آن هر فروشنده باید یکی از مقاصد را مورد بازدید قرار دهد.[۴۱] برای MDMTSP تبدیلی ارائه شد که در آن فروشنده‌ها از دو انبار شروع به حرکت می‌کردند.[۴۰] باید توجه داشت که برای نسخه‌ی MDMTSP هر فروشنده لازم نیست به انبار محل شروع حرکت خود برگردد و دست‌کم باید یکی از مقاصد را ملاقات کند. گیوخینگ[۱۴] یک تبدیل برای این مسئله به مسئله‌ی TSP نامتقارن ارائه کرد.[۴۲] در حال حاضر هیچ تبدیلی برای MDMTSP وجود ندارد که در آن بازگشت فروشنده به انبار محل شروع حرکت، به‌گونه‌یی که فقط در مسئله‌ی دو انبار وجود داشته باشد، ضرورت یابد.

الگوریتم رقابت بهره‌جویانه (ICA)[۱۵] یکی از جدیدترین الگوریتم‌های فرا ابتکاری است که تاکنون درمورد مسائل گوناگون بهینه‌سازی ترکیباتی پیوسته مورد استفاده قرار گرفته و به نتایج بسیار خوبی انجامیده است. از طرف دیگر چنان که گفته شد به‌علت اهمیت زیاد MTSP در مسائل تحقیق در عملیات و به‌خصوص مسئله‌ی مسیریابی وسیله‌ی نقلیه،[۴۳] در این نوشتار به این مسئله پرداخته می‌شود و برای اولین بار الگوریتم ICA برای آن پیشنهاد می‌شود که از کارایی بسیار خوبی برخوردار است.

در این مقاله در بخش ۲، مدل MTSP مورد بررسی قرار می‌گیرد و در بخش ۳ به توصیف روش ICA پرداخته می‌شود. در بخش ۴ روش پیشنهادی برای حل MTSP مورد بررسی قرار می‌گیرد. نتایج محاسباتی که درمورد مسائل استاندارد اجرا شده در بخش ۵ بررسی می‌شود. در نهایت نتیجه‌گیری و جهت‌گیری‌های آینده در بخش ۶ ارائه می‌شود.

۲. مدل MTSP

مسئله‌ی MTSP را می‌توان یک گراف غیر جهت‌دار کامل $G = (V, A)$ در نظر گرفت که در آن $V = \{۰, ۱, ..., n\}$ و $A = \{(i, j) : i, j \in V, i \neq j\}$



شکل ۱. یک جواب مسئله‌ی MTSP با $m = ۳$ و $n = ۱۲$.

به‌ترتیب مجموعه گره‌ها و مجموعه کمان‌ها را نشان می‌دهند. در صورت لزوم می‌توان گراف را با افزودن یال‌های مجازی با هزینه‌ی بسیار زیاد به یک گراف کامل تبدیل کرد. همچنین اگر در جواب نهایی (شکل ۱) برای $(i, j = ۰, ۱, ۲, ..., n; i \neq j)$ فروشنده به‌طور مستقیم از گره i به گره j حرکت کند آنگاه $x_{ij} = ۱$، و در غیر این صورت $x_{ij} = ۰$ در نظر گرفته می‌شود. به‌طور مثال اگر شکل ۱ یک جواب شدنی مسئله باشد، آنگاه مقادیر $x_{۰۴}, x_{۴۱۰}, ..., x_{۸۰}$ برابر ۱ هستند. در این مسئله اگر $C = [c_{ij}]$ نشان‌دهنده‌ی ماتریس هزینه روی گراف G باشد، آنگاه مدل برنامه‌ریزی صفر و ۱ مسئله عبارت است از:

$$\min \sum_{i=۰}^{n} \sum_{j=۰}^{n} c_{ij} x_{ij} \qquad (۱)$$

$$\sum_{i=۰}^{n} x_{ij} = \begin{cases} ۱ & j = ۱, ..., n \\ m & j = ۰ \end{cases} \qquad (۲)$$

$$\sum_{j=۰}^{n} x_{ij} = \begin{cases} ۱ & i = ۱, ..., n \\ m & i = ۰ \end{cases} \qquad (۳)$$

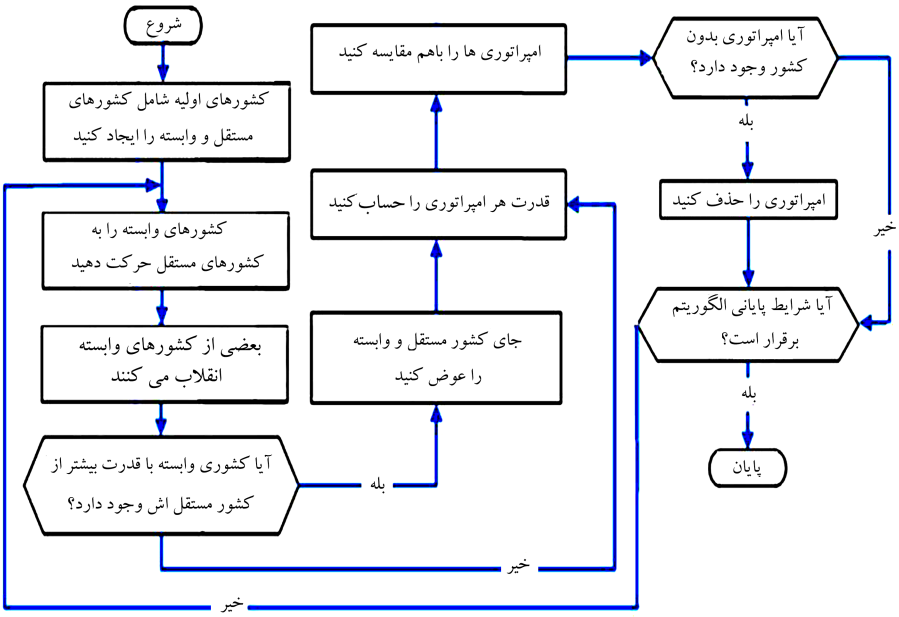
$$\sum\nolimits_{i,j \in S} x_{ij} \leq |S| - ۱ \quad (S \subset \{۱, ..., n\}, |S| \geq ۲) \qquad (۴)$$

$$x_{ij} \in \{۰, ۱\} \qquad i, j = ۰, ..., n \qquad (۵)$$

در این مدل، دسته محدودیت ۲ نشان می‌دهد که به هر گره j به غیر از انبار فقط یک یال، و به گره انبار m یال وارد می‌شود. همچنین دسته محدودیت ۳ مانند محدودیت دوم به این نکته اشاره دارد که از هر گره i به‌غیر از انبار فقط یک یال خارج می‌شود در حالی که این تعداد برای گره انبار m است. دسته محدودیت ۴ سبب می‌شود که الگوریتم از جواب‌هایی با یک زیر دور بدون انبار، به‌عنوان جواب قابل قبول،چشم‌پوشی کند. به‌عبارت دیگر، دسته محدودیت‌های ۲ و ۳ سبب می‌شوند که هر گره (غیر از انبار) فقط یک‌بار، و گره انبار به تعداد فروشنده‌ها مورد ملاقات قرار گیرند. بنابراین تنها حالتی که در آن ممکن است جواب به دست آمده شدنی نباشد آن است که دوری یافت شود که شامل گره انبار نباشد. محدودیت ۴ سبب می‌شود که این حالت نیز در عمل اتفاق نیفتد. سرانجام دسته محدودیت ۵ به شرایط دودویی متغیر اشاره می‌کند.

۳. الگوریتم رقابت بهره‌جویانه

امروزه مسائل بهینه‌سازی از اهمیت بسیار زیادی برخوردار است. این‌گونه مسائل در زندگی روزمره بسیارند و حل آن‌ها سبب حل مشکلات اجتماعی ـ اقتصادی می‌شود.



شکل ۲. فلوچارت الگوریتم ICA.

به‌علاوه این مسائل در اندازه‌های کاربردی و عملی توسط روش‌های بهینه‌سازی دقیق قابل حل نیستند و نمی‌توان در یک زمان قابل قبول به جواب بهینه‌ی آن‌ها دست یافت. از طرف دیگر از الگوریتم ابتکاری نمی‌توان برای تولید جواب‌های باکیفیت استفاده کرد؛ زیرا این‌گونه مسائل به‌علت پیچیدگی دارای تعداد زیادی بهینه‌های محلی هستند و الگوریتم‌های ابتکاری نمی‌توانند از این بهینه‌های محلی دور شوند. بنابراین محققین به دنبال الگوریتم‌هایی هستند که راهکاری مناسب برای دور شدن از نقاط بهینه محلی داشته باشند و بتوانند در یک زمان قابل قبول به جوابی باکیفیت دست یابند.

روش‌های فرابتکاری می‌توانند به این نیاز پاسخ داده و در یک زمان قابل قبول به جوابی مناسب دست یابند. این الگوریتم‌ها هنگامی‌که در یک بهینه‌ی محلی گیر می‌افتند، می‌توانند به سایر جواب‌های فضای شدنی دست پیدا کنند و مناطق بیشتری را مورد بررسی قرار دهند. بنابراین الگوریتم‌های فرابتکاری با جست‌وجوی بیشتر و استفاده از روالی کاراتر نسبت به روش‌های ابتکاری به جواب‌های بهتری دست پیدا می‌کنند.

الگوریتم‌های تکاملی دسته‌یی بسیار مهم از الگوریتم‌های فرابتکاری هستند که تاکنون برای بسیاری از مسائل بهینه‌سازی ترکیباتی ارائه شده‌اند.[۴۴] این الگوریتم‌ها با استفاده از فرایند تکامل طبیعی با یک جمعیت اولیه شروع می‌شوند و نسل به نسل با ارتقای جواب‌ها سعی می‌کنند که مناطق مهم ناحیه‌ی شدنی را مورد بررسی قرار دهند و به جواب‌های خوب همگرا شوند. الگوریتم‌های تکاملی زیادی تاکنون برای حل مسائل بهینه‌سازی ارائه شده که از آن جمله می‌توان به روش‌های ژنتیک، بهینه‌سازی گروهی ذرات و تکامل فرهنگی[۱۶] اشاره کرد.

الگوریتم ICA یکی از جدیدترین الگوریتم‌های تکاملی است که بر روند تکامل اجتماعی انسانی مبتنی است و در سال ۲۰۰۷ ارائه شد.[۴۵] این الگوریتم برخلاف بسیاری از روش‌های تکاملی که از طبیعت الهام گرفته‌اند، مبتنی بر یک فرایند اجتماعی ـ سیاسی به‌نام پدیده‌ی استعمار و بهره‌جویی است. به‌همین علت به‌عنوان نخستین الگوریتم بهینه‌سازی مبتنی بر یک فرایند اجتماعی ـ سیاسی محسوب می‌شود و از این جهت در نوع خود یک الگوریتم تکاملی جدید است.[۴۶]

از نظرکارایی نیز این الگوریتم تاکنون با موفقیت در تعدادی از مسائل بهینه‌سازی مورد استفاده قرارگرفته است. به‌عنوان مثال می‌توان به مسائل خوشه‌بندی داده‌ها،[۴۷] موتورالقایی خطی،[۴۸] تعادل نش[۴۹] و غیره اشاره کرد. استقبال بالا از این الگوریتم علاوه بر کارایی و سرعت همگرایی آن بیشتر ناشی از جنبه‌ی نوآوری و جذابیت آن برای متخصصین حوزه‌ی بهینه‌سازی است. این الگوریتم با تعدادی جمعیت اولیه شروع می‌شود که در اینجا «کشور» نامیده می‌شوند (شکل ۲). در مرحله‌ی بعد، این کشورها که جواب‌های شدنی برای مسئله‌ی مورد نظر هستند مقادیر قدرت (تابع هدف) خود را می‌یابند. با مقایسه‌ی این مقادیر برای کشورها می‌توان تعدادی از آن‌ها را که قدرت بیشتری دارند انتخاب و به‌عنوان کشورهای مستقل در نظر گرفت؛ سایر کشورها به‌عنوان کشورهای وابسته معرفی می‌شوند. حال هر کشور مستقل، بسته به توان خود، تعدادی از کشورهای وابسته را پوشش داده و حوزه‌ی نفوذ خود را تشکیل می‌دهد. باید توجه داشت که هرچه قدرت یک کشور مستقل بیشتر باشد کشورهای وابسته‌ی بیشتری را مورد پوشش قرار می‌دهد.

به‌علاوه کشورهای مستقل با دگرگونی فرهنگ و رسوم کشور وابسته سعی در تضعیف آن می‌کنند و همین سبب می‌شود که کشور وابسته در جهت همسان شدن فرهنگی با کشور مستقل حرکت کند. باید توجه داشت که در ارائه‌ی الگوریتم، این سیاست با حرکت دادن کشورهای وابسته به سمت کشوری مستقل در یک امپراتوری مطابق تابع جذب انجام می‌شود؛ این امر سبب افزایش قدرت کشورهای وابسته تا حد امکان می‌شود. توجه به این نکته ضروری است که در مقایسه با الگوریتم ژنتیک، این الگوریتم ترکیب کروموزوم‌ها را به‌صورت تصادفی انجام نداده و از تابع جذب به‌صورت منطقی استفاده می‌کند. به‌عبارت دیگر در این روش کشورهای وابسته با کشوری ترکیب می‌شوند که قدرت بیشتری دارند و نقش بهینه‌های محلی را بازی می‌کنند. در نتیجه ارتقاء کیفی جواب‌های حاصل از اعمال این روش بسیار محتمل‌تر است.

حرکت کشورهای وابسته به سمت کشورهای مستقل سبب افزایش کیفیت این کشورها می‌شود، اما پس از مدتی کشورها شباهت زیادی به یکدیگر پیدا می‌کنند و جواب‌ها با سرعت کم‌تری ارتقا می‌یابند. بنابراین لازم است که در این مرحله راهکاری

ارائه شود تا تعداد جواب‌های به دست آمده از تنوع بیشتری برخوردار شود. به‌همین علت درصدی (مثلاً ρ) ازکشورهای وابسته به‌وسیله‌ی یک الگوریتم محلی دچار تحول می‌شوند.

بعد از این که این دو عمل ــ تابع جذب و تحول برای هر امپراتوری ــ انجام شد ممکن است کشور وابسته به موقعیت بهتری نسبت به کشور مستقل خود دست یابد. در این صورت جای کشور وابسته و مستقل با هم عوض می‌شود. بنابراین با استفاده از تابع جذب و تحول روند جست‌وجوی سراسری انجام، و سعی می‌شود الگوریتم حرکت بهتری را در فضای ممکن انجام دهد و بتواند مناطق وسیع‌تری را با دقت بیشتر مورد بررسی قرار دهد.

روند منطقی موجود در سایر روش‌های فراابتکاری این است که الگوریتم باید به سمت مناطق مناسب و با کیفیت حرکت کند تا از حالت جست‌وجوی سراسری به جست‌وجوی محلی تبدیل شود. اکنون این سؤال مطرح می‌شود که چگونه مناطق مهم به‌وسیله‌ی این الگوریتم تعیین شوند تا بتوان به سمت آن‌ها حرکت کرد؟

برای این‌منظور باید قدرت هر منطقه را سنجید تا مناطق مهم با دقت بیشتر شناسایی شوند. در این الگوریتم قدرت کل یک امپراتوری به‌عنوان مجموع قدرت کشور مستقل به‌اضافه‌ی درصدی از میانگین قدرت کشورهای وابسته به آن تعریف می‌شود. بنابراین مناطق مهم بیشتری شناسایی می‌شود تا کشورهایی که از شانس کم‌تری برای ارتقای بیشتر در امپراتوری‌های خود برخوردارند به دیگر مناطق مهم‌تر بروند و با احتمال بیشتری رشد کنند. بدین‌ترتیب ضعیف‌ترین کشور در ضعیف‌ترین امپراتوری انتخاب شده و به یک امپراتوری دیگر ملحق می‌شود. به این ترتیب امپراتوری‌های ضعیف به‌تدریج قدرت خود را از دست می‌دهند و به مرور زمان از بین می‌روند. در نتیجه حالتی به دست می‌آید که در آن فقط یک امپراتوری وجود دارد که همه‌ی کشورها را اداره می‌کند و این زمانی است که الگوریتم ICA متوقف می‌شود.

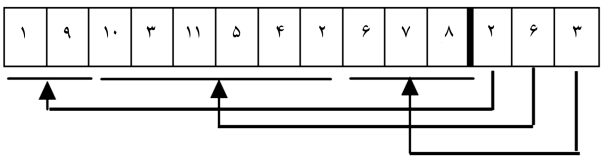
عمل حذف ضعیف‌ترین کشور از ضعیف‌ترین امپراتوری و الحاق آن به امپراتوری‌های دیگر بدان معناست که الگوریتم سعی می‌کند به سمت جست‌وجوی محلی متمایل شود. بنابراین در این الگوریتم طی مراحل متوالی، کشورها به سمت مناطقی راهنمایی می‌شوند که قدرت بیشتری دارند؛ این مناطق در مراحل بعدی با دقت بیشتری جست‌وجو می‌شوند.

۴. روش پیشنهادی

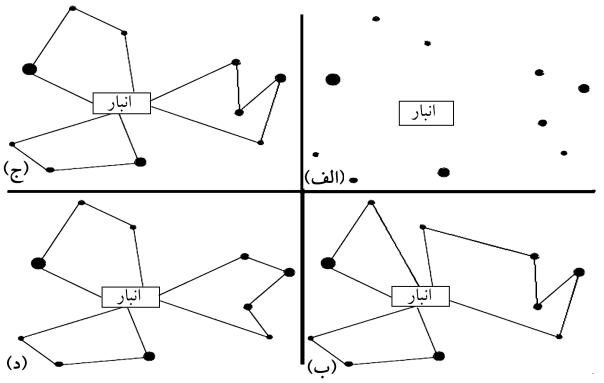
در این بخش مراحل الگوریتم پیشنهادی مورد بررسی قرار می‌گیرد:

مرحله‌ی ۱. برای این که بتوان الگوریتم پیشنهادی را درمورد MTSP اجرا کرد باید جواب‌های شدنی یا همان کشورهای اولیه را طوری معرفی کرد که با ساختار مسئله هماهنگی لازم را داشته باشد. بدین منظور از یک آرایه‌ی دوبخشی که در شکل ۳ نشان داده شده استفاده می‌شود. این آرایه به‌گونه‌یی است که گره‌ها به‌ترتیب ملاقات شدن از سمت چپ به راست در قسمت اولیه‌ی آرایه مرتب شده‌اند و سپس در قسمت دوم آرایه تعداد گره‌های ملاقات شده توسط هر فروشنده نشان داده شده است. به‌عبارت دیگر مجموع درایه‌های قسمت دوم آرایه برابر تعداد فروشنده‌هاست. به‌طور مثال در شکل ۳ ابتدا دو گره به‌وسیله‌ی فروشنده‌ی اول، شش گره به‌وسیله‌ی فروشنده‌ی دوم، و سه گره به‌وسیله‌ی فروشنده‌ی سوم ملاقات می‌شوند.

در الگوریتم پیشنهادی به تعداد معین و قابل تغییر P، جواب اولیه‌ی تصادفی ایجاد می‌شود و مقادیر تابع هدف f_i برای هر $i = ۱, ..., P$ محاسبه شود. سپس این جواب‌ها به‌همراه مقادیر آن‌ها در ماتریس D قرار می‌گیرند که در آن هر سطر



شکل ۳. یک کشور در الگوریتم ICA.



شکل ۴. مراحل حل مسئله به‌وسیله‌ی الگوریتم ICA.

نشان‌گر یک جواب اولیه‌ی ممکن به‌همراه تابع هدف آن‌هاست. استفاده از یک ساختار تصادفی در این مرحله سبب می‌شود که جواب‌های حاصله دارای ساختار متنوع در فضای ممکن باشند.

مرحله‌ی ۲. در این مرحله با مقایسه‌ی مقادیر تابع هدف برای هرکدام ازکشورها، I کشور که دارای مقدار تابع هدف بهتری هستند (در اینجا چون مسئله کمینه‌سازی است بنابراین کم‌ترین مقدار تابع هدف در نظر گرفته می‌شوند)، به‌عنوان کشورهای مستقل در نظرگرفته می‌شوند. با جابه‌جایی کشورها در ماتریس مربوطه، اندیس‌های ۱ تا I جمعیت اولیه در ماتریس D به کشورهای مستقل تعلق می‌یابد. برای تعیین حوزه‌ی نفوذ هرکشور مستقل، کشورهای وابسته طبق رابطه‌ی ۶ به کشورهای مستقل اختصاص می‌یابد تا بدین طریق I امپراتوری ایجاد شود.

$$k_j = \text{int}\left(\frac{۱/f_j}{\sum_{i=۱}^{I} ۱/f_i}.(P-I)\right) \qquad j = ۱, ..., I \qquad (۶)$$

رابطه‌ی ۶ نشان‌گر تعداد کشورهای وابسته‌ی اختصاص‌یافته به کشور مستقل jام است. افزون برا این، این رابطه سبب می‌شود که هرکدام ازکشورهای مستقل که تابع هدف بهتری دارد، تعداد کشورهای وابسته‌ی بیشتری به آن اختصاص یابد و امپراتوری بزرگ‌تری تشکیل شود. باید توجه کرد که «int» تابع جزء صحیح است که سبب می‌شود به هرکدام از امپراتوری‌ها تعداد صحیحی از کشورهای وابسته اختصاص یابد. از طرف دیگر ممکن است در خاتمه به‌علت استفاده از تابع int، کشورهایی وجود داشته باشند که به هیچ‌کدام از امپراتوری‌ها اختصاص نیابند؛ در این صورت این کشورها به کشور مستقل دارای بهترین مقدار تابع هدف، اختصاص خواهند یافت.

مرحله‌ی ۳. بعد از تشکیل امپراتوری‌ها، کشورهای هرامپراتوری با استفاده ازکشورهای مستقل که عهده‌دار نقش بهینه‌های محلی‌اند، کیفیت خود را افزایش می‌دهند (شکل ۴ب نشان‌دهنده‌ی یک کشور وابسته است). نکته‌ی قابل توجه این است که چون تعداد زیادی ازکشورهای وابسته با بهینه‌های محلی ترکیب می‌شوند پس باید از یک نوع تابع جذبی استفاده کرد که مفهوم تصادف در آن وجود داشته باشد تا جواب‌هایی مشابه تولید نشوند. بدین‌منظور از روش نزدیک‌ترین همسایه‌ی اصلاحی استفاده می‌شود. برای مثال، دو جواب شدنی [۵ ۳ ۳ | ۱۰ ۱۱ ۷ ۹ ۸ ۶ ۱ ۳ ۴ ۲ ۵] به

عنوان کشور مستقل و کشور [۵ ۴ ۲ | ۴ ۱۰ ۸ ۶ ۹ ۱۱ ۱ ۳ ۵ ۶ ۲] به عنوان کشور وابسته را در نظر بگیرید. حال از ۲ که اولین گره کشور وابسته است شروع به حرکت کنید و دو کشور همسایه‌ی ۲ را ــ یعنی ۴، ۵ و ۶ که تاکنون ملاقات نشده‌اند ــ در نظرگرفته و آن‌ها را در مجموعه‌ی S بریزید (اگر همسایه‌یی ملاقات نشده وجود نداشت آنگاه هرگره تاکنون ملاقات‌نشده را در S قرار دهید).

طبق روش نزدیک‌ترین همسایه‌ی اصلاحی احتمال ملاقات هرگره $j \in S$ درگره i برابر با ρ_{ij} است که از رابطه‌ی ۷ به دست می‌آید و در آن c_{ij} فاصله‌ی اقلیدسی گره i با گره j است. فرض کنید ۴ برگزیده شود پس در جمعیت اولیه تاکنون [۵ ۴ ۲ | ــ ــ ــ ــ ــ ــ ــ ــ ۴ ۲] وجود دارد. به‌علاوه برای ۴، چهارگره ۱، ۲، ۳ و ۱۰ به‌عنوان دو کشور همسایه محسوب می‌شود اما چون قبلاً ۲ مورد ملاقات قرارگرفته از بین ۱، ۳ و ۱۰ که مجموعه‌ی S جدید را تشکیل می‌دهند، نزدیک‌ترین همسایه‌ی احتمالی انتخاب می‌شود. این عمل تا یافتن بقیه‌ی گره‌ها در [۵ ۴ ۲ | ــ ــ ــ ــ ــ ــ ــ ــ ۴ ۲] و برای تمامی کشورهای وابسته ادامه می‌یابد. باید توجه داشت که در این روش تمامی گره‌ها فقط یک‌بار انتخاب می‌شوند، لذا جواب به دست آمده حتماً یک جواب ممکن است.

$$\rho_{ij} = \frac{1/c_{ij}}{\sum_{k \in S} 1/c_{ik}} \qquad (۷)$$

در همین مرحله با تحول ρ درصد ازکشورهای وابسته، امپراتوری‌های دارای تنوع کافی می‌شوند (شکل ۴ج). چون در مرحله‌ی قبلی، قسمت اول جواب‌ها مورد بررسی قرار گرفته و طبق الگوریتم نزدیک‌ترین همسایه‌ی احتمالی ارتقا یافته‌اند، در این مرحله قسمت دوم بردار جواب‌های شدنی مورد بررسی و اصلاح قرار می‌گیرد. برای این‌منظور یکی از درایه‌ها به تصادف انتخاب و یک واحد به آن اضافه می‌شود. برای مثال جواب شدنی [۵ ۴ ۲ | ۴ ۱ ۷ ۳ ۵ ۶ ۱۰ ۹ ۸ ۱۱ ۲] را در نظر بگیرید. فرض کنید ۴ از قسمت دوم انتخاب می‌شود و یک واحد به آن اضافه می‌شود. حال برای این‌که جواب به یک جواب ممکن تبدیل شود همسایه‌های ۴، یعنی ۲ و ۵ را در نظر بگیرید. چون یک واحد به ۴ اضافه شده، بنابراین یک واحد از یکی از همسایه‌ها به‌طور تصادفی کم می‌شود. به‌طور مثال از ۵ یک واحد کم می‌شود و جواب جدید [۴ ۵ ۲ | ۴ ۱ ۷ ۳ ۵ ۶ ۱۰ ۹ ۸ ۱۱ ۲] به وجود می‌آید.

بعد از این‌که جواب و مقدار جدید برای ρ درصد ازکشورهای وابسته محاسبه شد، ممکن است این کشورها دارای قدرت بیشتری نسبت به کشور مستقل درگروه خود شوند. بنابراین کشوری وابسته با بهترین مقدار در هر امپراتوری انتخاب و با کشور مستقل مقایسه می‌شود. در صورتی که چند کشور وابسته دارای مقدار یکسان باشند، یکی از آن‌ها به تصادف انتخاب و با کشور مستقل امپراتوری خود مقایسه و در صورت لزوم جابه‌جا می‌شود.

مرحله‌ی ۴. تا انتهای تکرار الگوریتم دیگر تغییری در کشورها صورت نمی‌گیرد. بنابراین باید بهترین جواب و مقدار تابع هدف ذخیره شود. بدین منظور در هر تکرار، بعد از این‌که عمل جایگزینی کشورهای مستقل به‌وسیله‌ی کشورهای وابسته با مقادیر بهتر انجام شد، بهترین جواب در بین کشورهای مستقل به‌عنوان بهترین جواب جاری انتخاب می‌شود، و در صورتی که نسبت به جواب‌های قبلی بهتر باشد، جست‌وجوی محلی دوگانه روی آن اجرا می‌شود و جواب جدید جایگزین جواب و مقدار قبلی می‌شود.

این روش، همان‌طور که در شکل ۴ج و ۴د نشان داده شده است، براساس حذف دو یال از مسئله و جایگزین کردن آن با دو یال مناسب‌تر اجرا می‌شود. این تغییر زمانی پذیرفته می‌شود که جواب جدید بهتر از جواب قبلی باشد. باید توجه کرد که برای این جایگزینی حالت‌های مختلفی وجود دارد اما تنها حالتی پذیرفته می‌شود که در محدودیت‌های مسئله صدق کند.

تا این مرحله از الگوریتم، هدف جست‌وجوی سراسری برای یافتن مناطق مهم است. حال باید این مناطق مهم را به‌وسیله‌ی رابطه‌ی ۸ شناسایی کرد و جمعیت به سمت این مناطق همگرا شود. بنابراین در مرحله‌ی چهارم قدرت امپراتوری‌های را باید سنجید. این عمل به‌وسیله‌ی رابطه‌ی ۸ انجام می‌گیرد:

$$h_j = f_j - \lambda(s_j) \qquad j = 1, ..., I \qquad (۸)$$

که در آن h_j، f_j، s_j و λ به‌ترتیب نشان‌دهنده‌ی قدرت کل هر امپراتوری، قدرت هرکشور مستقل (مقدار تابع هدف)، میانگین قدرت کشورهای وابسته در امپراتوری jام، و ضریب تأثیر است. ضریب تأثیر عددی بین صفر و ۱ است که اهمیت قدرت کشورهای وابسته نسبت به کشور مستقل در یک امپراتوری را تعیین می‌کند. حال امپراتوری ضعیف‌تر، ضعیف‌ترین کشور وابسته‌ی خود را به قوی‌ترین امپراتوری می‌دهد. باید توجه کرد که اگر امپراتوری ضعیف‌تر دارای کشوری وابسته نباشد از بین می‌رود و خود به‌عنوان کشوری وابسته به قوی‌ترین امپراتوری اختصاص می‌یابد.

مرحله‌ی ۵. در این مرحله شرط پایانی الگوریتم مورد بررسی قرار می‌گیرد؛ در صورت تحقق شرط مذکور، الگوریتم به پایان می‌رسد و در غیر این صورت الگوریتم با برگشت به مرحله‌ی سوم دوباره تکرار می‌شود.

برای خاتمه‌ی الگوریتم از دو شرط تکرار الگوریتم به تعداد معین T بار یا باقی ماندن فقط یک امپراتوری استفاده می‌شود که به‌طور همزمان در پایان هر تکرار الگوریتم مورد بررسی قرار می‌گیرد. هرکدام از این دو شرط که زودتر اتفاق بیفتد الگوریتم به پایان می‌رسد و بهترین جواب و مقداری که تاکنون به دست آمده است به‌عنوان جواب نهایی مسئله معرفی می‌شود.

۵. نتایج محاسباتی

تمام کدهای این برنامه به زبان C نوشته شده است؛ رایانه‌یی که این برنامه‌ها بر روی آن اجرا شده از نوع AMD ۳GHz با یک گیگا بایت حافظه است. همچنین سیستم عامل نصب شده روی این رایانه ویندوز XP است. در این بخش نتایج به دست آمده به‌وسیله‌ی الگوریتم ICA نشان داده شده است. بدین‌منظور دو دسته از مسائل در نظرگرفته شده‌اند. دسته‌ی اول شامل ۶ مسئله و برگرفته از کتابخانه‌ی TSP است، و دسته‌ی دوم شامل ۲۱ مسئله است.[۵۰] اگرچه این دسته از مثال‌ها از نظر تعداد گره نسبت به دسته اول تنوع کم‌تری دارند اما از نظر تعداد فروشنده‌ها بازه مناسبی دارند و مسائلی از ۳ تا ۳۰ فروشنده را شامل می‌شوند.

با تعیین تعداد فروشنده‌ها برای هر TSP می‌توان آن‌ها را MTSPهای استاندارد تبدیل کرد. در این مسائل تعداد گره‌ها بین ۷۶ تا ۱۰۰۲ است. خصوصیات کامل دسته اول مسائل همراه با نتایج محاسباتی الگوریتم پیشنهادی در جدول ۱ ارائه شده است. ستون دوم این جدول نشان‌گر تعداد گره‌ها (n) و ستون سوم نشان‌گر تعداد فروشنده‌ها (m) در هر مسئله است. برای برقراری تعادل بین تعداد گره‌های ملاقات شده توسط هر فروشنده، عدد l به‌عنوان بیشترین تعداد گره‌های قابل ملاقات برای هر فروشنده تعیین شده است. tb در ستون پنجم نشان‌دهنده‌ی تعداد تکراری است که الگوریتم به‌طور مستقل انجام می‌دهد. T در ستون ششم نشان‌گر تعداد تکرار الگوریتم به‌عنوان یکی از شرایط خاتمه است و برابر تعداد گره هر مسئله در نظرگرفته می‌شود. سرانجام در ستون هفتم بهترین مقادیری که الگوریتم توانسته در tb تکرار مستقل به دست آورد، نشان داده شده است.

جدول ۱. خصوصیات مسائل استاندارد و نتایج محاسباتی.

خصوصیات مسائل مورد آزمایش				نتایج محاسباتی		
مسئله	n	m	l	tb	T	ICA
Pr۷۶	۷۶	۵	۲۰	۱۰	۷۶	۱۵۷۴۴۲
Pr۱۵۲	۱۵۲	۵	۴۰	۱۰	۱۵۲	۱۲۷۸۳۹
Pr۲۲۶	۲۲۶	۵	۵۰	۱۰	۲۲۶	۱۶۶۸۱۹
Pr۲۹۹	۲۹۹	۵	۷۰	۱۰	۲۹۹	۸۲۰۹۸
Pr۴۳۹	۴۳۹	۵	۱۰۰	۱۰	۴۳۹	۱۶۱۹۵۵
Pr۱۰۰۲	۱۰۰۲	۵	۲۲۰	۱۰	۱۰۰۲	۳۸۳۱۷۸

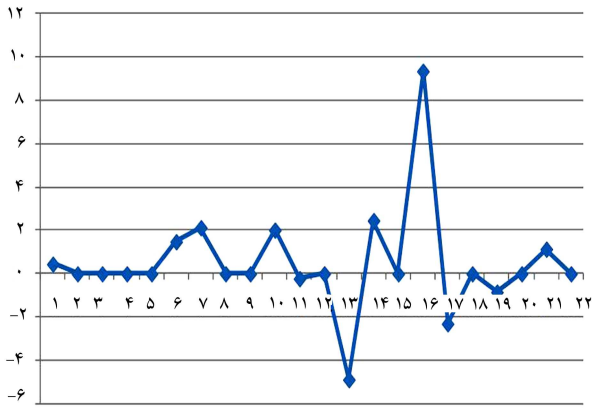
در جدول ۲ نتایج الگوریتم‌های فراابتکاری برای مسائل جدول ۱ گزارش شده است. درستون دوم نتایج الگوریتم ICA، و درستون‌های سوم و چهارم به‌ترتیب نتایج الگوریتم‌های اصلاحی ژنتیک (MGA)[۴۶] و اصلاحی مورچگان (MACO)[۵۱] از ادبیات موضوع درج شده است. باید توجه داشت که در این جدول هریک از ستون‌های MGA و MACO به سه زیرستون تقسیم می‌شوند. این سه ستون شامل بهترین مقدار، میانگین مقادیر و همچنین زمان اجرای الگوریتم در ۱۰ تکرار است. به‌علاوه ستون ICA چهار زیرستون دارد و علاوه بر موارد گفته شده بدترین جواب در ۱۰ تکرار را نیز نشان داده است. از طرف دیگر در ستون‌های پنجم و ششم به‌ترتیب بهترین جواب حاصله به‌وسیله‌ی همه‌ی الگوریتم‌ها (Best) و Gap الگوریتم پیشنهادی که به‌وسیله‌ی رابطه‌ی ۹ به دست آمده، ارائه شده است.

$$\text{Gap} = ۱۰۰ \times \frac{\text{Value of the Best} - \text{Value of the ICA}}{\text{Value of the Best}} \qquad (۹)$$

با مقایسه‌ی جواب‌ها می‌توان نتیجه گرفت که الگوریتم پیشنهادی نسبت به MGA به جواب‌های خوبی دست یافته و در بیش از ۰٫۸۰ موارد کیفیت جواب‌ها را افزایش داده است. به‌علاوه دومین الگوریتمی که نتایج آن با روش پیشنهادی مقایسه شده روش MACO است. علی‌رغم این که این الگوریتم در دو مسئله‌ی Pr۴۳۹ و Pr۱۰۰۲ به جواب‌های بهتری دست یافته است، در چهار مسئله‌ی دیگر نتوانسته جواب‌های بهتری نسبت به الگوریتم پیشنهادی به دست آورد. بنابراین می‌توان نتیجه گرفت که الگوریتم ICA از قدرت بسیار خوبی برخوردار است و به‌طور کلی از نقاط بهینه‌ی محلی فرار کرده است. در مجموع می‌توان الگوریتم‌های جدول ۲ را براساس بهترین جواب به دست آمده، به‌ترتیب از ضعیف به قوی به‌صورت MGA، MACO و ICA مرتب کرد.

از طرف دیگر اگر ستون میانگین جواب‌های به دست آمده توسط هر الگوریتم در ۱۰ تکرار متوالی در جدول ۲ در نظر گرفته و سپس برای شش مثال مربوطه میانگین آنها محاسبه شود، آنگاه مقادیر ۱۸۳۲۸۵٫۷ برای ICA، ۲۰۰۱۸۰٫۸ برای MGA و ۱۸۷۳۳۲٫۲ برای MACO به دست می‌آید. از این نظر نیز الگوریتم ICA بهترین جواب‌ها را ارائه کرده است.

جدول ۳ نشان‌دهنده‌ی مشخصات ۲۱ مسئله است. این مسائل دارای ۵۱، ۱۰۰ و ۱۵۰ گره هستند. در این جدول و در ستون دوم، m نشان‌دهنده‌ی تعداد فروشنده‌هاست. به‌علاوه چهار الگوریتم GGA، GA۲PC، GGA-SS و ICA در ستون‌های سوم تا ششم نشان داده شده است. سه الگوریتم اول دارای دو ستون هستند که ستون اول میانگین به دست آمده به‌وسیله‌ی روش مربوطه، و ستون دوم میانگین زمان اجرای الگوریتم برحسب ثانیه در ۱۰ تکرار الگوریتم است. از طرف دیگر دو ستونِ بهترین جواب و بدترین جواب، علاوه بر دو ستون گفته شده میانگین و زمان، برای الگوریتم ICA وجود دارد. لازم به ذکر است که جواب‌های گزارش شده در روش GGA-SS به‌وسیله‌ی رایانه‌یی با مشخصات پنتیوم ۴ با قدرت



شکل ۵. نتایج Gap میانگین الگوریتم ICA در مقایسه با بهترین میانگین جواب‌های به دست آمده.

جدول ۲. مقایسه‌ی الگوریتم‌های فراابتکاری.

مسئله	ICA				MGA			MACO			Best	Gap
	بهترین	بدترین	میانگین	زمان	بهترین	میانگین	زمان	بهترین	میانگین	زمان		
Pr۷۶	۱۵۷۴۴۲	۱۶۷۳۴۲	۱۵۹۲۳۱	۶٫۲۴	۱۵۷۴۴۴	۱۶۰۵۷۴	۴۳	۱۷۸۵۹۷	۱۸۰۶۹۰	۵۱	۱۵۷۴۴۴	+۰٫۰۰
Pr۱۵۲	۱۲۷۸۳۹	۱۳۴۸۷۷	۱۳۰۷۱۲	۱۲٫۸۷	۱۲۷۸۳۹	۱۳۳۳۳۷	۹۱	۱۳۰۹۵۳	۱۳۶۳۴۱	۱۲۸	۱۲۷۸۳۹	۰٫۰۰
Pr۲۲۶	۱۶۶۸۱۹	۱۷۵۳۶۱	۱۷۰۹۲۳	۲۰٫۶۲	۱۶۶۸۲۷	۱۷۸۵۰۱	۱۶۵	۱۶۷۶۴۶	۱۷۰۸۷۷	۱۴۳	۱۶۶۸۲۷	+۰٫۰۰
Pr۲۹۹	۸۲۰۹۸	۸۵۱۱۰	۸۳۸۱۲	۲۱٫۹۷	۸۲۱۷۶	۸۵۷۹۶	۳۶۳	۸۲۱۰۶	۸۳۸۴۵	۲۸۸	۸۲۱۰۶	+۰٫۰۱
Pr۴۳۹	۱۶۱۹۵۵	۱۷۲۸۶۱	۱۶۷۹۰۷	۶۵٫۸۱	۱۷۳۸۳۹	۱۸۳۶۹۸	۶۲۳	۱۶۱۹۵۵	۱۶۵۰۳۵	۵۶۳	۱۶۱۹۵۵	۰٫۰۰
Pr۱۰۰۲	۳۸۳۱۷۸	۳۹۲۳۱۴	۳۸۷۱۲۹	۲۳۴٫۶۵	۴۲۷۲۶۹	۴۵۹۱۷۹	۲۸۹۲	۳۸۲۱۹۸	۳۸۷۲۰۵	۲۶۲۰	۳۸۲۱۹۸	−۰٫۲۶

جدول ۳. مقایسه‌ی الگوریتم پیشنهادی با نسخه‌های مختلف الگوریتم ژنتیک.

مسئله	m	GGA [۵۲]		GA۲PC [۵۰]		GGA-SS [۵۳]		ICA				Gap
		میانگین	زمان	میانگین	زمان	میانگین	زمان	بهترین	بدترین	میانگین	زمان	
MTSP – ۵۱	۳	۹۲۴	۳۰۰	۵۴۳	۳۰۰	۴۴۹	۳٫۹۴	۴۴۷	۴۴۷	۴۴۷	۴٫۹۵	۰٫۴۵
	۵	۸۸۲	۳۰۰	۵۸۶	۳۰۰	۴۷۹	۳٫۳۸	۴۷۳	۴۸۵	۴۷۹	۴٫۲۳	۰٫۰۰
	۱۰	۱۰۰۱	۳۰۰	۷۲۳	۳۰۰	۵۸۴	۳٫۵۴	۵۸۳	۵۸۸	۵۸۴	۵٫۰۷	۰٫۰۰
MTSP – ۱۰۰ – I	۳	۷۹۳۴۷	۶۰۰	-	-	۲۲۱۰۰	۶٫۲۲	۲۲۱۰۰	۲۲۱۰۰	۲۲۱۰۰	۹٫۱۵	۰٫۰۰
	۵	۷۰۸۷۱	۶۰۰	-	-	۲۳۳۹۸	۸٫۵۶	۲۳۳۹۸	۲۳۳۹۸	۲۳۳۹۸	۸٫۸۹	۰٫۰۰
	۱۰	۸۹۷۷۸	۶۰۰	-	-	۲۸۳۵۶	۶٫۴۲	۲۷۵۹۳	۲۸۲۳۱	۲۷۹۴۲	۷٫۵۴	۱٫۴۸
	۲۰	۱۳۷۸۰۵	۶۰۰	-	-	۴۱۵۵۴	۷٫۰۲	۳۹۳۲۱	۴۱۱۲۱	۴۰۶۸۸	۸٫۱۲	۲٫۱۳
MTSP – ۱۰۰ – II	۳	-	-	۲۶۶۵۳	۶۰۰	۲۲۰۵۱	۶٫۳۳	۲۲۰۵۱	۲۲۰۵۱	۲۲۰۵۱	۷٫۹۰	۰٫۰۰
	۵	-	-	۳۰۴۰۸	۶۰۰	۲۳۶۷۸	۸٫۰۵	۲۳۶۷۸	۲۳۶۷۸	۲۳۶۷۸	۹٫۱۱	۰٫۰۰
	۱۰	-	-	۳۱۲۲۷	۶۰۰	۲۸۴۸۸	۶٫۴۵	۲۷۷۰۲	۲۸۷۶۵	۲۷۹۲۵	۶٫۴۵	۲٫۰۲
	۲۰	-	-	۵۴۷۰۰	۶۰۰	۴۰۸۹۲	۷٫۶۸	۳۹۷۴۲	۴۲۳۱۱	۴۰۹۸۵	۸٫۷۳	−۰٫۲۳
MTSP – ۱۵۰ – I	۳	۳۳۸۸۸	۹۰۰	-	-	۶۶۳۲	۲۴٫۰۲	۶۶۳۲	۶۶۳۲	۶۶۳۲	۱۰٫۸۹	۰٫۰۰
	۵	۲۶۸۵۱	۹۰۰	-	-	۶۷۵۱	۱۵٫۵۲	۶۸۴۳	۷۱۹۲	۷۰۹۸	۸٫۴۳	−۴٫۸۹
	۱۰	۳۷۷۷۱	۹۰۰	-	-	۷۸۸۵	۷٫۷۲	۷۵۹۹	۷۷۹۰	۷۶۹۶	۹٫۶۵	۲٫۴۶
	۲۰	۴۳۶۹۹	۹۰۰	-	-	۱۰۳۹۹	۸٫۲۶	۹۸۰۱	۱۰۵۵۱	۱۰۳۹۹	۹٫۶۱	۰٫۰۰
	۳۰	۵۲۵۶۴	۹۰۰	-	-	۱۴۹۲۹	۸٫۰۵	۱۳۲۷۵	۱۳۸۵۶	۱۳۶۵۲	۱۱٫۲۴	۹٫۳۵
MTSP – ۱۵۰ – II	۳	-	-	۴۷۴۱۸	۹۰۰	۳۸۴۳۴	۱۵	۳۸۵۳۱	۳۹۹۹۳	۳۹۳۴۲	۲۵٫۳۲	−۲٫۲۱
	۵	-	-	۴۹۹۴۷	۹۰۰	۳۹۹۶۲	۱۱٫۵۸	۳۹۹۶۲	۳۹۹۶۲	۳۹۹۶۲	۱۱٫۰۹	۰٫۰۰
	۱۰	-	-	۵۴۹۵۸	۹۰۰	۴۴۲۷۴	۹٫۳	۴۴۱۳۳	۴۴۸۷۰	۴۴۶۵۲	۱۰٫۱۱	−۰٫۸۵
	۲۰	-	-	۷۳۹۳۴	۹۰۰	۵۶۴۱۲	۱۱٫۱۱	۵۵۵۹۳	۵۸۲۰۱	۵۶۴۱۲	۱۲٫۲۳	۰٫۰۰
	۳۰	-	-	۹۹۵۴۷	۹۰۰	۷۲۷۸۳	۶٫۹	۷۰۹۲۳	۷۲۳۳۱	۷۱۹۶۵	۱۳٫۶۷	۱٫۱۴

3GHz و با ۵۱۲ مگابایت حافظه به دست آمده است. در حالی که مشخصات رایانه الگوریتم ICA از نوع پنتیوم با قدرت ۳ GHz و با ۱ گیگا بایت حافظه است.

با مقایسه‌ی مقادیر به دست آمده می‌توان نتیجه گرفت که الگوریتم ICA کارایی خوبی برای حل این مسائل دارد زیرا توانسته است نسبت به دو الگوریتم GGA و GA۲PC در تمامی مسائل مورد آزمایش جواب‌های بهتری برای میانگین جواب‌ها کسب کند. به‌علاوه این الگوریتم در مقایسه با الگوریتم GGA-SS نیز توانسته در ۷ مورد از ۲۱ مسئله‌ی مورد آزمایش، کیفیت میانگین جواب‌ها را افزایش دهد و در ۱۰ مورد به جواب‌های یکسانی دست پیدا کند.

برای آگاهی از کیفیت میانگین الگوریتم پیشنهادی در شکل ۵ نتایج آن در مقایسه با بهترین میانگین جواب‌های تاکنون به دست آمده نشان داده شده است. به‌وضوح در این شکل مشاهده می‌شود که الگوریتم توانسته جواب‌های بسیار خوبی به دست آورد و فقط در ۴ مثال است که نتوانسته کیفیت جواب‌ها را افزایش دهد.

از طرف دیگر چون مشخصات رایانه‌های مورد استفاده در الگوریتم پیشنهادی و الگوریتم GGA-SS بسیار به هم شبیه‌اند می‌توان نتیجه گرفت که زمان اجرا در هر دو الگوریتم تقریباً یکسان است.

۶. نتیجه‌گیری

در این مقاله الگوریتم جدید فراابتکاری ICA برای حل MTSP مورد استفاده قرار گرفت. در مقایسه با سایر الگوریتم‌های فراابتکاری، روش پیشنهادی، به‌خصوص در مسائل با اندازه بزرگ، برخوردار بوده و می‌تواند تا حد ممکن از بهینه‌های محلی فرار کند. به نظر می‌رسد کاربرد این روش برای دیگر مسائل بهینه‌سازیِ ترکیباتی، نظیر مسئله‌ی مسیریابی وسیله‌ی نقلیه یا گسترش‌های آن، می‌تواند نتایج خوبی در بر داشته باشد. از طرف دیگر چون کارایی الگوریتم برای یافتن جواب مسائل خیلی بزرگ است، ترکیب این الگوریتم با دیگر روش‌های فراابتکاری مانند الگوریتم مورچگان، ژنتیک، شبیه‌سازی آنیلی یا جست‌وجوی ممنوع راهکاری مناسب برای بهبود آن و یافتن جواب‌هایی باکیفیت‌تر است.

پانوشت‌ها

1. multiple traveling salesmen problem (MTSP)
2. traveling salesman problem (TSP)
3. vehicle routing problem (VRP)
4. single depot multiple traveling salesmen problem (SDMTSP)
5. multiple depot multiple traveling salesmen problem (MDMTSP)
6. multiple traveling salesmen problem with time window (MTSPTW)
7. vehicle scheduling problem (VSP)
8. overnight security service problem
9. printing press scheduling
10. Yadlapalli
11. Gromicho
12. Russell
13. hot rolling problem
14. GuoXing
15. imperialist competitive algorithm (ICA)
16. cultural algorithm

منابع (References)

1. Park, Y.B. "A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines", *International Journal of Productions Economics*, **73**(2), pp. 175-188 (2001).
2. Yousefikhoshbakht, M. and Rahmati, F."An improved ant colony optimization for solving the vehicle routing problem with simultaneous pickup and delivery", Transportation Research Journal, **8**(2), pp. 183-199 (in Persian) (2011).
3. Taghavifard, M., Sheikh, K. and Shahsavari, A. "Modified ant colony algorithm for the vehicle routing problem with time windows", *International Journal of Industrial and Production Management,* **20** (2), pp.23-30 (in Persian)(2009).
4. Calvo., R.W. and Cordone, R. "A heuristic approach to the overnight security service problem", *Computers and Operations Research,* **30**, pp. 1269-1287 (2003).
5. Carter, A.E. and Ragsdale, C.T. "Scheduling pre-printed newspaper advertising inserts using genetic algorithms", *Omega*, **30**(6), pp. 415-421 (2002).
6. Gorenstein, S. "Printing press scheduling for multi-edition periodicals", *Management Science,* **16**(6), pp. 73-83 (1970).
7. Svestka, J.A. and Huckfeldt, V.E. "Computational experience with an m-salesman traveling salesman algorithm", *Management Science,* **19**(7), pp. 790-799 (1973).
8. Angel, R.D., Caudle, W.L., Noonan, R. and Whinston, A. "Computer assisted school bus scheduling", *Management Science*, **18**, pp. 279-288 (1972).
9. Gilbert, K.C. and Hofstra, R.B. "A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem", *Decision Sciences,* **23**, pp. 250-259 (1992).
10. Brummit, B. and Stentz, A. "Dynamic mission planning for multiple mobile robots", *Proceedings of the IEEE International Conference on Robotics and Automation* (1996).
11. Brummit, B. and Stentz, A. "GRAMMPS: A generalized mission planner for multiple mobile robots", *Proceedings of the IEEE International Conference on Robotics and Automation* (1998).
12. Tang, L., Liu, J., Rong, A. and Yang, Z.A. "multiple traveling salesman problem model for hot rolling scheduling in shangai baoshan Iron & Steel complex", *European Journal of Operational Research,* **124**, pp. 267-282 (2000).
13. Saleh, H.A. and Chelouah, R. "The design of the global navigation satellite system surveying networks using genetic algorithms", *Engineering Applications of Artificial Intelligence,* **17**, pp. 111–122 (2004).
14. Bektas, T. "The multiple traveling salesman problem: An overview of formulations and solution procedures", *Omega*, **34**, pp. 209-219 (2006).
15. Finke, G., Claus, A. and Gunn, E.A. "Two-commodity network flow approach to the traveling salesman problem", *Congressus Numerantium*, **41**, pp. 167-178 (1984).
16. Miliotis, P. "Using cutting planes to solve the symmetric travelling salesman problem", *Mathematical Programming,* **15**(1), pp. 177-188 (1978).
17. Bhide, S., John, N. and Kabuka, M.R. "A boolean neural network approach for the liaveling salesman problem", *IEEE Transactions on Computers*, **42**(10), pp. 1271-1278 (1993).
18. Glover, F. "Artificial intelligence, heuristic frameworks and tabu search", *Managerial and Decision Economics*, **11**(5), pp. 365-375 (1990).
19. Yadlapalli, S., Malik, W.A., Darbhaa, S. and Pachter, M. "A Lagrangian-based algorithm for a multiple depot, multiple traveling salesmen problem", *Nonlinear Analysis: Real World Applications*, **10**(4), pp. 1990-1999 (2009).
20. Gromicho, J., Paixão., J. and Branco, I. "Exact solution of multipletraveling salesman problems", In: MustafaAkgül, et al., editors. Combinatorial Optimization. NATO ASI Series, F82, Berlin: Springer; pp. 291-92 (1992).
21. Laporte, G. and Nobert, Y.A. "Cutting planes algorithm for the m-salesmen problem", *Journal of the Operational Research Society*, **31**, pp. 1017-1023 (1980).
22. Saad, S., Jaafar, W.N.W. and Jamil, S.J., "Solnny standard trareling salesman problem and multiple traveling salesman problem by using branch and bound", *AIP Conference Proceeding,Malaysia* (in Persian)(2013).
23. Ali, A. and Kennington, J.L. "Exact solution of multiple traveling salesman problems", *Discrete Applied Mathematics,* **13**, pp. 259-276 (1986).
24. Gavish, B. and Srikanth, K. "An optimal solution method for large-scale multiple traveling salesman problems", *Operations Research*, **34**(5), pp. 698-717 (1986).
25. Potvin, J., Lapalme, G. and Rousseau, J. "A generalized k-opt exchange procedure for the MTSP", *INFOR*, **21**, pp. 474-481 (1989).
26. Russell, R.A. "An effective heuristic for the m-tour traveling salesman problem with some side conditions", *Operations Research,* **25**(3), pp. 517-524 (1977).

27. Potvin, J., Lapalme, G. and Rousseau, J. "A generalized k-opt exchange procedure for the MTSP", *INFOR*, **21**, pp. 474-481 (1989).
28. Qu, H., Yi, Z. and Tang, H. "A columnar competitive model for solving multi-traveling salesman problem", *Chaos, Solitons and Fractals,* **31**, pp. 1009-1019 (2005).
29. Hsu, C., Tsai, M. and Chen, W. "A study of feature-mapped approach to the multiple travelling salesmen problem", *IEEE International Symposium on Circuits and Systems*, **3**, pp. 1589-1592 (1991).
30. Ryan, J.L., Bailey, T.G., Moore, J.T. and Carlton, W.B. "Reactive Tabu search in unmanned aerial reconnaissance simulations", *Proceedings of the 1998 Winter Simulation Conference* (1998).
31. Zhang, T., Gruver, W.A. and Smith, M.H. "Team scheduling by genetic search", *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials* (1999).
32. Gen, M. and Cheng, R., *Genetic Algorithms and Engineering Design*, Wiley- Interscience (1997).
33. Zhang, T., Gruver, W.A. and Smith, M.H. "Team scheduling by genetic search", *Proceedings of the Second International Conference on Intelligent Processing and Manufacturing of Materials,* **2**, pp. 839-844 (1999).
34. Malmborg, C. "A genetic algorithm for service level based vehicle scheduling", *European Journal of Operational Research,* **93**(1), pp. 121-134 (1996).
35. Park, Y.B. "A hybrid genetic algorithm for the vehicle scheduling problem with due times and time deadlines", *International Journal of Productions Economics,* **73**(2), pp. 175-188 (2001).
36. Tang, L., Liu, J., Rong, A. and Yang, Z. "A multiple traveling salesman problem model for hot rolling scheduling in shangai baoshan iron & steel complex", *European Journal of Operational Research*, **124**, pp. 267-282 (2000).
37. Yu, Z., Jinhai, L., Guochang, G., Rubo, Z. and Haiyan, Y. "An implementation of evolutionary computation for path planning of cooperative mobile robots", *Proceedings of the Fourth World Congress on Intelligent Control and Automation,* **3**, pp. 1798-1802 (2002).
38. Bellmore, M. and Hong, S.A. "Note on the symmetric Multiple Travelling Salesman Problem with fixed charges", *Operations Research,* **25**, pp. 871-874 (1977).
39. Hong, S. and Padberg, M.W. "A note on the symmetric multiple traveling salesman problem with fixed charges", *Operations Research,* **25**(5), pp. 871-874 (1977).
40. Rao, M.R. "A note on multiple travelling salesmen problem", *Operations Research,* **28**(3), pp. 628-632 (1980).
41. Jonker, R. and Volgenant, T. "An improved transformation of the symmetric multiple traveling salesman problem", *Operations Research*, **36**(1), pp. 163-167 (1988).
42. GuoXing, Y. "Transformation of multi-depot multi-salesmen problem to the standard traveling salesman problem", *European Journal of Operations Research,* **81**, pp. 557-560 (1995).
43. Zafari, A., Tashakori, S. and Yousefikhoshbakht, M. "A hybrid effective genetic algorithm for solving the vehicle routing problem", *International Journal of Industrial and Production Management,* **21** (2), pp. 63-76 (in Persian)(2010).
44. Bozorg Haddad, O., Afshar, A. and Afshar, H. "Honey-bee mating optimization (HBMO) algorithm in optimization problems", it International Journal of Industrial and Production Management, **19** (2), 99-112 (in Persian)(2008).
45. Atashpaz-Gargari, E. and Lucas, C. "Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition", In: *Proceedings of the IEEE Congress on Evolutionary Computation*, Singapore (2007).
46. Kaveh, A. and Talatahari, S. "Optimum design of skeletal structures using imperialist competitive algorithm", *Computers & Structures*, **88**(21-22), pp. 1220-1229 (2010).
47. Niknam, T., TaherianFard, E., Pourjafarian., N. and Rousta, A. "An efficient hybrid algorithm based on modified imperialist competitive algorithm and K-means for data clustering", *Engineering Applications of Artificial Intelligence,* **24**(2), pp. 306-317 (2011).
48. Lucas, C., Nasiri-Gheidari, Z. and Tootoonchian, F. "Application of an imperialist competitive algorithm to the design of a linear induction motor", *Energy Conversion and Management*, **51**(7), pp. 1407-1411 (2010).
49. Rajabioun, R., Atashpaz-Gargari, E. and Lucas, C. "Colonial competitive algorithm as a tool for nash equilibrium point achievement", Lecture Notes in Computer Science, LNCS 5073, pp. 680-695 (2008).
50. Carter, A.E. and Ragsdale, C.T. "A new approach to solving the multiple traveling salesperson problem using genetic algorithms", *European Journal of Operational Research,* **175**, pp. 246-257 (2006).
51. Junjie, P. and Dingwei, W. "An ant colony optimization algorithm for multiple travelling salesman problem", In ICICIC '06: *Proceedings of the First International Conference on Innovative Computing*, Information and Control, Washington, DC, USA, IEEE Computer Society (2006).
52. Brown, E.C., Ragsdale, C.T. and Carter, A.E. "A grouping genetic algorithm for the multiple traveling salesperson problem", *International Journal of Information Technology & Decision Making,* **6**(2), pp. 333-347 (2007).
53. Singh, A. and Baghel, A.S. "A new grouping genetic algorithm approach to the multiple traveling salesperson problem", *Soft Computing*, **13**, pp. 95-101 (2009).