International Journal of Industrial Engineering & Production Management (2016)

December 2016, Volume 27, Number 4
pp. 671-682

http://IJIEPM.iust.ac.ir/

A binary Imperialist Competitive Algorithm to Semi-obnoxious Facility Location Models on Networks with Subgraph-shaped Customers

Behrooz Alizadeh*, Ahmad Lotfi

Behrooz Alizadeh, *Department of Applied Mathematics, Sahand University of Technology*
Ahmad Lotfi, *Department of Applied Mathematics, Sahand University of Technology*

Keywords

Optimal facility location,
Binary imperialist competitive algorithm,
Subgraph-shaped customers,
NP-hard models

ABSTRACT

In a median location problem on networks (graphs), the aim is to find the best locations for establishing the facilities on the underlying graph such that the sum of weighted distances between customers and the facilities is minimized. In this paper, we consider a semi-obnoxious (pos/neg weighted) p-median location model on general networks in which the existing customers are given as 'subgraphs'. A novel binary imperialist competitive algorithm is proposed in order to obtain the solutions of the problem under investigation. Our computational tests show that this algorithm runs with a high acceleration.

نشـــریه بین المللـــی مهنـــدسی صنـــایع و مدیـــریت تولیـــد

شمـــاره ۴، جلـــد ۲۷، زمستان ۱۳۹۵
صفحـــه ۶۸۲-۶۷۲

نشریه بین المللی مهندسی صنایع و مدیریت تولید
دانشگاه علم و صنعت ایران

ISSN: 2008-4870

http://IJIEPM.iust.ac.ir/

الگوریتم رقابت استعماری دودویی برای مدل‌های مکان‌یابی تسهیلات نیمه ناخوشایند روی شبکه‌ها با مشتریان به شکل زیرگراف

بهروز علی‌زاده*، احمد لطفی

کلمات کلیدی

مکان‌یابی بهینه تسهیلات، الگوریتم رقابت استعماری دودویی، مشتریان به شکل زیرگراف، مدل‌های NP-سخت

چکیده:

در یک مسئله مکان‌یابی میانه[1] روی شبکه‌ها (گراف‌ها)، هدف پیدا کـردن یـک مجموعـه از بهتـرین مکان‌ها روی یک گراف معین جهت تاسیس یا استقرار تسهیلات مـی‌باشـد بـه طـوری کـه مجمـوع فواصل مشتریان موجود از نزدیک‌ترین سرویس‌دهنده کمینه شود. در این مقاله، یک مدل مکان‌یـابی $p-$ میانه نیمه ناخوشایند[2] روی شبکه‌ها در نظر گرفته می‌شود که در آن تمامی مشتریان بـه شـکل زیرگراف تعریف می‌گردند. یک الگوریتم رقابت استعماری دودویی جدید برای یافتن جواب‌های مـدل تحت مطالعه پیشنهاد می‌گردد. نتایج محاسباتی ما نشان می‌دهند که این الگوریتم از سرعت و دقت بالایی برخوردار می‌باشد.

۱. مقدمه

مسائل مکان‌یابی تسهیلات (سرویس‌دهندگان) جزو مدل‌های بهینه‌سازی در حوزه تحقیق در عملیات می‌باشند که با توجه به کاربردهای فراوانی که در تئوری و عمل دارند، همواره مورد توجه محققان قرار گرفته‌اند. به عنوان کاربردهای مدل‌های مکان‌یابی می‌توان به پیدا کردن مکان‌های بهینه مراکز پستی، بیمارستان-ها، فرودگاه‌ها، ایستگاه‌های آتش‌نشانی، پایگاه‌های نظامی، نیروگاه‌های اتمی، پایانه‌های شبکه اتوبوس‌رانی درون‌شهری [۱]، مراکز تسهیلات حساس [۲] و غیره اشاره کرد [۳]،[۴].

یک نوع متداول از مسائل مکان‌یابی، مدل مکان‌یـابی $p-$ میانـه می‌باشد که در آن هدف این است که یک مجموعـه از بهتـرین مکان‌ها برای تأسیس p سرویس‌دهنده را روی سیستم تحت مطالعه (شبکه یا فضای حقیقـی چنـدبعـدی) پیـدا کنـیم بـه طوری که مجموع فواصل (وزن‌دار) مشتریان موجود از نزدیـک-ترین سرویس دهنده تاسیس شده بـه حـداقل برسـد. حکیمـی نشان داد که روی گراف‌ها، جستجوی یـک جـواب بهینـه بـرای مسئله مکان‌یابی $p-$ میانه می‌تواند به مجموعه رأس‌های گـراف محدود گردد که این ویژگی به *خاصیت بهینگی رأسی*[1] معـروف گردیده است [5]. در سال۱۹۷۹، کریو و حکیمی نشـان دادنـد که مدل مکان‌یابی $p-$ میانه روی گراف‌هـا از لحـاظ پیچیـدگی NP- سخت[2] می‌باشد [6]. بعداً بورکارد و همکاران [7] دو نـوع مسئله $p-$ میانه نیمه ناخوشایند ($p-$ میانه با وزن‌هـای رأسـی مثبت/منفی) روی گراف‌ها را برای اولین بار مـورد بررسـی قـرار داده و الگوریتم‌های ترکیبیاتی برای یـافتن جـواب‌هـای بهینـه ارائه نمودند. آن‌ها نشان دادند که خاصیت بهینگی رأسی بـرای مدل‌های $p-$ میانه نیمه ناخوشایند روی گراف‌ها در حالت کلـی برقرار نمی‌باشد.

در مسائل دنیای واقعی در برخی موارد با وضعیتی مواجه هستیم که نمی‌توان مکان هر مشتری موجود در سیستم تحت مطالعه را به صورت یک *نقطه* تعریف کرد. به همین دلیل، در

تاریخ وصول: ۹۴/۰۲/۲۹
تاریخ تصویب: ۹۴/۱۰/۰۵

احمد لطفی، گروه ریاضـی کـاربردی، دانشـگاه صـنعتی سـهند، تبریـز، ایـران، alotfi36@yahoo.com

***نویسنده مسئول مقاله: بهروز علی‌زاده**، گروه ریاضـی کـاربردی، دانشـگاه صنعتی سهند، تبریز، ایران، alizadeh@sut.ac.ir

سال‌های اخیر مکان‌یابی تسهیلات با مشتریان به فرم *زیرگراف*[۳] همواره مورد توجه محققان قرار گرفته است. در سال ۲۰۰۸، پیورتو و همکاران [8] مسائل مکان‌یابی $p-$مرکز روی گراف‌های درختی با مشتریان به شکل زیردرخت را در نظر گرفته و رویکردهای مبتنی بر مدل‌های پوششی با زمان اجرای چندجمله‌ای برای به دست آوردن جواب‌های بهینه ارائه نمودند.

چنگ وکانگ [9] مشتریان به شکل زیردرخت را مورد مطالعه قرار داده و یک الگوریتم جواب با پیچیدگی زمانی خطی پیشنهاد کردند. در سال ۲۰۱۰، زانگ و همکاران [10] نشان دادند که مسئله بالا می تواند روی گراف‌های بلوکی با مشتریان به شکل زیرگراف در زمان خطی حل گردد. در این مقاله، یک مدل مکان‌یابی $p-$میانه نیمه ناخوشایند روی " گراف‌های کلی" با مشتریان به شکل زیرگراف را به صورت یک مدل برنامه‌ریزی خطی صفر-یک فرمول‌بندی کرده و یک الگوریتم رقابت استعماری دودویی برای یافتن جواب مسئله پیشنهاد می‌شود.

سازماندهی مقاله به صورت زیر می باشد: در بخش۲، مدل مکان‌یابی مورد نظر را بیان کرده و یک مدل برنامه‌ریزی خطی صفر-یک معادل برای آن فرمول‌بندی می‌شود. یک الگوریتم رقابت استعماری دودویی برای یافتن جواب‌های مطلوب مسئله در بخش ۳ پیشنهاد می‌گردد. بخش ۴ به ارائه نتایج محاسباتی و تجزیه و تحلیل آن‌ها می‌پردازد.

۲. بیان مسئله و مدل برنامه‌ریزی خطی صفر-یک معادل

فرض کنید $G=\big(V(G),\,E(G)\big)$ یک شبکه (گراف) همبند با مجموعه رأسی $V(G)=\{v_1,...,v_n\}$ و مجموعه یالی $E(G)$ باشد. برای هر یال $e\in E(G)$، یک طول مثبت $l(e)$ تخصیص داده می‌شود. فرض کنید یک مجموعه τ عضوی $\Lambda=\{G_i : i=1,...,\tau\}$ از زیرگراف‌های G، به عنوان مجموعه محل‌های مشتریان موجود داده شده باشد. برای هر مشتری G_i یک وزن $w_i\in\mathbb{R}$ به عنوان اهمیت نسبی آن اختصاص داده می‌شود. اگر $d^l(x,y)$ نشان‌دهنده طول کوتاه‌ترین مسیر بین دو نقطه $x,y\in V(G)$ برحسب طول‌های یالی شبکه باشد، آنگاه فاصله بین یک رأس $x\in V(G)$ و یک زیرگراف $G'\subseteq G$ به صورت زیرتعریف می‌شود:

$$D^l(x,G')=\min\left\{d^l(x,y) \mid y\in V(G')\right\}. \qquad (1)$$

در یک مدل مکان‌یابی $p-$میانه نیمه ناخوشایند روی شبکه G با مشتریان به شکل زیرگراف، هدف این است که یک مجموعه $X_p^*\subseteq V(G)$، $|X_p^*|=p$، را به عنوان محل تاسیس p سرویس‌دهنده روی G پیدا کنیم به طوری که یک جواب بهینه برای مدل بهینه‌سازی غیرخطی زیر باشد:

$$\begin{aligned}&\text{minimize} \quad \sum_{i=1}^{\tau}\min_{1\le j\le p}\left(w_i\, D^l(x_j,G_i)\right) &&(2)\\ &\text{subject to} \quad X_p=\{x_1,...,x_p\}\subseteq V(G).\end{aligned}$$

قضیه زیر را داریم:

قضیه ۱-۲. یک مدل مکان‌یابی $p-$میانه نیمه ناخوشایند روی شبکه‌ها با مشتریان به شکل زیرگراف از لحاظ پیچیدگی NP−سخت است.

اثبات: یک نمونه خاص از مسئله را روی یک گراف G در نظر بگیرید که در آن $\tau=n$ بوده و مشتریان زیرگرافی به صورت $G_i=v_i\,,\, i=1,...,\tau,$ تعریف شوند. فرض کنید به هر مشتری G_i یک وزن غیر منفی w_i تخصیص داده شود. این نمونه خاص، یک مسئله مکان‌یابی $p-$میانه با "مشتریان نقطه‌ای" می‌باشد که در حالت کلی روی گراف‌ها NP−سخت می‌باشند [6]. لذا ادّعای قضیه نتیجه گرفته می‌شود.

در ادامه سعی داریم یک مدل برنامه‌ریزی خطی صفر-یک معادل برای مسئله مکان‌یابی (۲) فرمول‌بندی نماییم. به ازای هر $i=1,...,\tau$ و $j=1,...,n$، ابتدا متغیرهای تصمیم‌گیری $y_j,\,x_{ij}$ را به صورت زیر تعریف نمایید:

$$x_{ij}=\begin{cases}1 & \text{اگر مشتری } G_i \text{ به سرویس‌دهنده } v_j \text{ تخصیص یابد}\\ 0 & \text{در غیر این صورت}\end{cases}$$

و

$$y_j=\begin{cases}1 & \text{اگر رأس } v_j \text{ به عنوان مکان سرویس‌دهنده انتخاب شود}\\ 0 & \text{در غیر این صورت}\end{cases}$$

فرض کنید $[m]_i$، نشان‌دهنده mامین مکان سرویس‌دهنده نسبت به مشتری G_i باشد به طوری که $m\in\{1,2,...,n-1\}$. مسئله مکان‌یابی $p-$میانه نیمه ناخوشایند (۲) می‌تواند به صورت برنامه‌ریزی خطی صفر- یک زیر فرمول‌بندی شود:

$(\mathbf{BLP})$:

$$\begin{aligned}&\text{minimize} \quad \Gamma(x)=\sum_{i=1}^{\tau}\sum_{j=1}^{n} w_i\, D^l(v_j,G_i)x_{ij}\\ &\text{subject to} \quad \sum_{j=1}^{n} x_{ij}=1 \qquad ; i=1,...\tau, &&(3)\end{aligned}$$

$$-x_{ij}+y_j \geq 0 \quad ; i=1,\ldots\tau,\ j=1,\ldots,n, \qquad (۴)$$

$$\sum_{j=1}^{n} y_j = p, \qquad (۵)$$

$$-y_{[m]_i}+\sum_{k=1}^{m} x_{i,[k]_i} \geq 0 \quad ; m=1,\ldots,n-1, \qquad (۶)$$
$$\forall i:\ w_i<0,$$
$$y_j \in \{0,1\} \quad ; j=1,\ldots,n,$$
$$x_{ij} \in \{0,1\} \quad ; i=1,\ldots,\tau,\ j=1,\ldots,n.$$

محدودیت‌های (۳) تضمین می‌کنند که هر مشتری دقیقاً از یک سرویس‌دهنده خدمت‌رسانی شود. دسته محدودیت‌های (۴) بیانگر امکان دریافت خدمت توسط هرمشتری G_i ازمکان v_j تنها در صورت تاسیس سرویس‌دهنده در آن محل می‌باشد. محدودیت (۵) احداث دقیقاً p سرویس‌دهنده را تضمین می کند. محدودیت‌های (۶) کنترل می‌کنند که هر مشتری از نزدیک‌ترین سرویس‌دهنده، خدمت‌رسانی شود.
همان طور که در قضیه ۱-۲ ذکر شد، مدل مکان‌یابی $p-$میانه نیمه ناخوشایند با مشتریان به شکل زیرگراف روی شبکه‌ها از رده $NP-$سخت می‌باشد. مروری بر ادبیات تحقیقی نشان می‌دهد که تاکنون هیچ الگوریتم چندجمله‌ای دقیق برای مدل‌های بهینه‌سازی $NP-$سخت ارائه نشده است. از طرفی الگوریتم‌های دقیق غیر چندجمله‌ای نیز در اکثر اوقات قادر نمی‌باشند که در مقیاس بزرگ این نوع مسائل سخت را در یک زمان قابل قبول حل نمایند. لذا طراحی و بکارگیری الگوریتم‌های فراابتکاری برای مدل‌های $NP-$سخت از اهمیت خاصی برخوردار است. بنابراین، یک الگوریتم رقابت استعماری دودویی (BICA) جدید با دقت و سرعت همگرایی بالا جهت به دست آوردن جواب مدل BLP معادل در بخش بعدی ارائه خواهد شد.

۳. الگوریتم رقابت استعماری

۳-۱. الگوریتم رقابت استعماری پایه‌ای

الگوریتم رقابت استعماری یک روش جستجوی فراابتکاری با ساختار تصادفی برمبنای تکامل اجتماعی- سیاسی انسان‌ها می‌باشد که توسط آتش‌پز و لوکاس **[11]** برای اولین بار در سال ۲۰۰۷ میلادی معرفی گردید. اگر چه مدت چندان زیادی از ابداع این الگوریتم نمی گذرد، اما تاکنون برای حل مدل‌های بهینه‌سازی مختلفی مانند طراحی چیدمان سیستم‌های تولید انعطاف‌پذیر [۱۲]، جایابی پوششی هاب ظرفیت‌دار سرویس دهندگان **[13]**، مسئله فروشنده دوره‌گرد **[14]** و غیره از این الگوریتم استفاده شده است. برای یک مطالعه مروری در خصوص الگوریتم رقابت استعماری و کاربردهای آن در حل مسائل علوم مهندسی می‌توانید به **[15]** مراجعه نمایید.
این الگوریتم با یک جمعیتی از کشورها در گام آغازین شروع می‌شود به طوری که این کشورها به دو دسته تقسیم می‌شوند: کشورهای امپریالیست و کشورهای مستعمره. در واقع رقابت امپریالیستی هسته اصلی این الگوریتم را تشکیل داده و باعث می‌شود که کلیه کشورها به قوی‌ترین امپریالیست موجود جذب گردند. این امر از دیدگاه ریاضیاتی به معنی همگرا شدن دنباله جواب‌های شدنی مدل بهینه‌سازی تحت مطالعه به یک جواب بهینه است. همانند دیگر الگوریتم‌های تکاملی، این الگوریتم نیز با تعدادی جمعیت اولیه تصادفی که هر کدام از آن‌ها یک کشور نامیده می‌شود، شروع می‌گردد. تعدادی از بهترین عناصر به عنوان کشورهای امپریالیست انتخاب شده و باقی جمعیت نیز به عنوان کشورهای مستعمره در نظر گرفته می‌شوند. استعمارگران بسته به قدرت‌شان، این مستعمرات را با یک روند خاص به سمت خود جذب می‌نمایند. قدرت کل هر امپراطوری به هر دو بخش تشکیل‌دهنده آن، یعنی قدرت کشور امپریالیست و قدرت مستعمرات آن بستگی دارد
با شکل‌گیری امپراطوری‌های اولیه، رقابت امپریالیستی میان آن‌ها شروع می‌شود. هر امپراطوری که نتواند در رقابت استعماری موفق عمل کرده و بر قدرت خود بیفزاید و یا حداقل از کاهش نفوذش جلوگیری کند، از صحنه رقابت استعماری حذف خواهد شد. در نتیجه در جریان رقابت‌های امپریالیستی به تدریج بر قدرت امپراطوری‌های بزرگ‌تر افزوده شده و امپراطوری‌های ضعیف‌تر، حذف خواهند شد. امپراطوری‌ها برای افزایش قدرت خود، مجبور خواهند شد که قدرت و موقعیت مستعمرات خود را ارتقاء دهند. به این ترتیب، با گذشت زمان، کشورهای مستعمره از لحاظ قدرت به امپراطوری‌ها نزدیک شده و یک نوع همگرایی بوجود می‌آید. شرط همگرایی در صورت وجود امپراطوری واحد همراه با مستعمراتی که از لحاظ موقعیت به کشور امپریالیست خود خیلی نزدیک هستند، حاصل خواهد شد. در ادامه، مراحل مختلف الگوریتم رقابت استعماری با جزئیات کافی بیان می‌گردد.
شکل‌گیری امپراطوری‌های اولیه: در یک الگوریتم رقابت استعماری برای حل یک مدل بهینه سازی N_{var} بعدی نوع کمینه‌سازی، یک کشور، یک بردار $1\times N_{var}$ بعدی به فرم $\Pi=\left[\pi_1,\pi_2,\ldots,\pi_{N_{var}}\right]$ می باشد. در حقیقت $\pi_1,\pi_2,\ldots,\pi_{N_{var}}$ نشان‌دهنده همان متغیرهای تصمیم مدل بهینه‌سازی تحت مطالعه هستند. هزینه متناظر هر کشور Π، با ارزیابی تابع هدف (تابع هزینه) Γ مدل

بهینه‌سازی مورد نظر به ازای مقادیر متغیرهـای $\pi_1,\pi_2,...,\pi_{N_{var}}$ محاسبه می‌گردد، به‌عبارتی دیگر

$$\text{cost}=\Gamma(\Pi)=\Gamma(\pi_1,\pi_2,...,\pi_{N_{var}}).$$

برای شروع اجـرای الگـوریتم، تعـداد $N_{country}$ کشـور بـه عنـوان جمعیت اولیه و به صورت تصادفی ایجاد می‌شوند به طوری کـه اگر مدل بهینه‌سازی از نوع مقید باشد، انگـاه هـر کـدام یـک از این کشورهای اولیه باید در محدودیت‌های مسئله بهینـه‌سـازی صدق نماید. تعداد N_{imp} بهترین عضو این جمعیـت (کشـورهای دارای کمترین هزینه متناظر) به عنوان کشورهای امپریالیسـت انتخاب شده و تعداد $N_{col}=N_{country}-N_{imp}$ کشور باقیمانده در جمعیت، مستعمراتی را تشکیل می‌دهند که هر کـدام بـه یـک کشور امپریالیست تعلق پیدا می‌کند. بنابراین، مستعمرات اولیـه بین کشورهای امپریالیست اولیه تقسیم می‌شوند به طـوری کـه تعداد مستعمرات اختصاص یافته به هر امپریالیست متناسب بـا قدرت آن امپریالیست می باشد. برای انجام این کار، بـا موجـود بودن هزینه متناظر همه کشورهای امپریالیست، هزینـه نرمـال شده امپریالیست Π_{imp_s}، $1\le s\le N_{imp}$، به صورت زیـر محاسـبه می‌گردد:

$$C_{imp_s}=\max_{1\le i\le N_{imp}}\{\text{cost}_{imp_i}\}-\text{cost}_{imp_s}$$

توجه کنید که هرکشور امپریالیستی کـه دارای هزینـه متنـاظر بیش‌تری باشد (امپریالیست ضـعیف‌تـری باشـد)، دارای هزینـه نرمال شده کمتری خواهد بود. بعـد از محاسـبه هزینـه نرمـال شده تمامی امپریالیسـت‌هـای Π_{imp_j}، قـدرت نرمـال شـده هـر امپریالیست به صورت زیر محاسبه شده و سپس بر مبنـای آن، کشورهای مستعمره بین امپریالیست‌ها توزیع می‌شوند:

$$\varphi_s=\left|\frac{C_{imp_s}}{\sum_{j=1}^{N_{imp}}C_{imp_j}}\right|,\qquad 1\le s\le N_{imp}.$$

از دیدگاهی دیگر، قدرت نرمال شده یـک امپریالیسـت، نسـبت مستعمراتی است کـه توسـط آن امپریالیسـت اداره مـی‌شـوند. بنابراین تعداد مستعمرات اولیه امپریالیست Π_{imp_s} برابر با

$$N_{col}^{s}=\text{round}(\varphi_s\times N_{col})$$

خواهد بود به طوری که

$$N_{col}=\bigcup_{j=1}^{N_{imp}}N_{col}^{j}.$$

تعداد N_{col}^{s} کشور مستعمره اولیه بـه صـورت تصـادفی انتخـاب شده و به امپریالیست Π_{imp_s} تخصیص داده می‌شوند. بـا شـکل-گیری امپراطوری‌های اولیه، یکی از عملیـات اساسـی الگـوریتم رقابت استعماری به صورت زیر انجام می‌پذیرد[11].

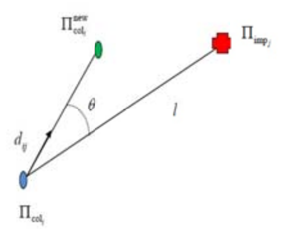
حرکت مستعمرات به سمت یک امپریالیست (سیاست جـذب): همان‌طورکه قبلاً اشـاره شـد، کشـورهای امپریالیسـت، موجـب رشد و افزایش قدرت کشورهای مستعمره خود مـی‌شـوند. ایـن فرآیند حقیقی به وسیله حرکت مسـتعمرات بـه سـمت کشـور امپریالیسـت مربوطـه خـود مـدل‌بنـدی مـی‌گـردد. حـال یـک مسـتعمره Π_{col_i} را در نظـر بگیریـد کـه قـرار اسـت بـه سـمت امپریالیست Π_{imp_j} حرکت نماید. فرض کنید d_{ij} بـردار واحـدی باشد که با خط واصل نقاط Π_{col_i} و Π_{imp_j} زاویـه θ بـر حسـب رادیان می سازد به طوری که θ یک عـدد تصـادفی بـا توزیـع یکنواخت در بازه $[-\gamma,\gamma]$ است، یعنی

$$\theta\sim U(-\gamma,\gamma).$$

γ یک پارامتر ورودی با مقداری دلخواه مـی باشـد کـه انـدازه انحراف مستعمره از جهت اصلی را نشان می‌دهد. مکـان جدیـد مستعمره Π_{col_i} به صورت

$$\Pi_{col_i}=\Pi_{col_i}+\delta d_{ij}$$

محاسبه می‌گردد (شـکل ۱ را ملاحظـه کنیـد) بـه طـوری کـه $\delta\sim U(0,\beta\times l)$ و β یک عدد حقیقـی دلخـواه بـزرگ‌تـر از **1** بوده و l نشان‌دهنده فاصله بین مستعمره Π_{col_i} و امپریالیسـت Π_{imp_j} است.



شکل ۱. حرکت مستعمرات به سمت امپریالیست مربوطه

جابجایی موقعیت یک امپریالیست با مستعمره خـود (انقـلاب): در جریـان فرآینـد حرکـت مسـتعمرات بـه سـمت کشـورهای امپریالیست، ممکن است یـک مسـتعمره Π_{col_i} تحـت اسـتعمار یک امپریالیست Π_{imp_j} مـوقعیتی بهتـر از امپریالیسـت مربوطـه خود پیدا نماید، بعبارتی

$$\Gamma(\Pi_{col_i})<\Gamma(\Pi_{imp_j}).$$

در این حالت، کشور امپریالیست Π_{imp_j} و کشور مستعمره Π_{col_i} جای خـود را بـا همـدیگر عـوض کـرده و اجـرای الگـوریتم بـا کشورهای امپریالیست جدید ادامه یافتـه و ایـن بـار کشـورهای

امپریالیست جدید می‌باشند که شروع به اعمال سیاست جذب بر مستعمرات خود می‌نمایند.

قدرت کل یک امپراطوری: قدرت کل یک امپراطوری از طریق مجموع قدرت کشور امپریالیست و ضریبی از میانگین قدرت مستعمرات آن امپراطوری محاسبه می‌شود. توجه به این نقطه ضروری است که این ضریب موجب می‌گردد که تصمیم‌گیرنده بتواند تأثیر اهمیت کشور امپریالیست یا میانگین کشورهای مستعمره را در قدرت نهایی هر امپراطوری تعیین کند. به عبارتی دیگر، اگر $\boldsymbol{I}_j$ نشان‌دهنده مجموعه اندیس‌های مربوط به کشورهای مستعمره امپریالیست Π_{imp_j} باشد، آنگاه قدرت امپراطوری Π_{imp_j} از طریق محاسبه هزینه کل آن، یعنی

$$C^{\text{total}}(\Pi_{\text{imp}_j}) = \cos t\left(\Pi_{\text{imp}_j}\right) + \zeta(\text{iter}) \times \left(\frac{\sum_{i \in I_j} \cos t\left(\Pi_{\text{imp}_i}\right)}{\left|\boldsymbol{I}_j\right|} \right) \quad (۷)$$

معین می‌گردد. در این مقاله، برای محاسبه $\zeta(\text{iter})$ ضابطه

$$\zeta(\text{iter}) = 0.01 \times \frac{\text{Maxit}}{\text{iter} + \text{Maxit}}$$

پیشنهاد می‌گردد که در آن iter، نشان‌دهنده شماره تکرار فعلی الگوریتم در طول اجرای آن بوده و Maxit نشان‌دهنده ماکزیمم تعداد تکرارهای مجاز از قبل تعیین شده توسط تصمیم‌گیرنده می‌باشد. در منابع دیگر، $\zeta(\text{iter})$ را یک عدد ثابت متعلق به بازه $(0 , 1)$ در طول اجرای الگوریتم در نظر گرفته‌اند. براساس آزمایشات انجام گرفته می‌توان گفت که فرمول پیشنهادی بالا برای $\zeta(\text{iter})$، دقت و سرعت الگوریتم ما را به نحو قابل قبولی افزایش می دهد.

رقابت استعماری: هر کشور امپریالیستی که نتواند بر قدرت خود بیفزاید، در جریان رقابت امپریالیستی، حذف خواهد شد. این عمل حذف به صورت تدریجی صورت می‌پذیرد. به این معنی که به مرور زمان امپراطوری‌های ضعیف، مستعمرات خود را از دست داده و امپراطوری‌های قوی‌تر، این مستعمرات را تصاحب کرده و بر قدرت خود می‌افزایند. به منظور فرموله‌سازی این واقعیت طبیعی، فرض می‌شود که کشور امپریالیست در حال حذف، ضعیف‌ترین امپراطوری موجود باشد. در طول تکرار الگوریتم رقابت استعماری، یک یا چند عضو از ضعیف‌ترین مستعمرات تحت سلطه ضعیف‌ترین امپراطوری را انتخاب کرده و سپس برای تصاحب این مستعمرات، رقابتی میان امپراطوری‌ها ایجاد می‌گردد. لازم به ذکر است که مستعمرات مذکور، لزوماً توسط قوی‌ترین امپراطوری تصاحب نخواهند شد. بنابراین، برای مدل‌سازی ریاضیاتی عمل رقابت میان امپراطوری‌های موجود جهت تصاحب مستعمرات انتخاب شده، ابتدا احتمال تصاحب هر امپراطوری باید محاسبه شود که بر مبنای قدرت آن امپراطوری می‌باشد. برای این منظور، ابتدا به ازای هر $1 \le j \le N_{\text{imp}}$، هزینه کل نرمال شده کشور امپریالیست Π_{imp_j} باید از طریق رابطه

$$N\_C^{\text{total}}_{\text{imp}_j} = \max_{1 \le s \le N_{\text{imp}}} \left\{ C^{\text{total}}(\Pi_{\text{imp}_s}) \right\} - C^{\text{total}}(\Pi_{\text{imp}_j})$$

محاسبه گردد. سپس احتمال استعمار یک کشور مستعمره توسط امپریالیست‌های Π_{imp_j} به صورت زیر

$$P_{\text{imp}_j} = \left| \frac{N\_C^{\text{total}}_{\text{imp}_j}}{\sum_{s=1}^{N_{\text{imp}}} N\_C^{\text{total}}_{\text{imp}_s}} \right| \quad , 1 \le j \le N_{\text{imp}},$$

تعیین گردیده و برای تقسیم کشورهای مستعمره بین امپریالیست‌ها، بردار احتمال

$$\text{Prob} = \left[P_{\text{imp}_1}, P_{\text{imp}_2}, \ldots, P_{\text{imp}_{N_{\text{imp}}}} \right] \quad (۸)$$

تشکیل می‌شود. در ادامه یک بردار تصادفی

$$\text{Rnd} = \left[r_1, r_2, \ldots, r_{N_{\text{imp}}} \right],\ r_1, \ldots, r_{N_{\text{imp}}} \sim U(0,1) \quad (۹)$$

را تولید کرده و بردار تخصیص

$$\Delta = \left[\Delta_1, \Delta_2, \ldots, \Delta_{N_{\text{imp}}} \right] = \text{Prob} - \text{Rnd}. \quad (۱۰)$$

باید محاسبه شود. مستعمرات مذکور به امپراطوری تخصیص داده می‌شوند که شاخص مربوط به آن در بردار Δ بزرگ‌تر از بقیه امپریالیست‌ها باشد. امپراطوری که بیش‌ترین احتمال استعمار را داشته باشد، با احتمال بیش‌تری شاخص مربوط به آن در بردار Δ دارای ماکزیمم مقدار خواهد بود.

سقوط امپراطوری‌های ضعیف: یک امپراطوری زمانی حذف شده تلقی می‌شود که تمامی کشورهای مستعمره خود را در رقابت استعماری از دست بدهد. این کار در طول اجرای الگوریتم در اثر حرکت مستعمرات و عوض شدن موقعیت آن‌ها با امپریالیست متناظر خود رخ می‌دهد.

همگرایی: در طی مراحل اجرای الگوریتم، همه امپراطوری‌های ضعیف در تکرارهای مختلف سقوط کرده و تنها یک امپراطوری باقی خواهد ماند به طوری که سایر کشورهای مستعمره همگی تحت نفوذ این امپراطوری بوده و دارای موقعیت و هزینه‌های یکسان خواهند بود. در این حالت می‌توان به اجرای الگوریتم خاتمه داد. البته معیارهای ختم دیگری نیز می‌تواند توسط تصمیم‌گیرنده تعریف شود.

با توجه به اینکه هدف ما یافتن جواب برای مدل برنامه ریزی خطی صفر-یک می باشد، لذا در ادامه یک الگوریتم رقابت استعماری دودویی جدید پیشنهاد خواهد شد که تعمیمی از الگوریتم رقابت استعماری پایه‌ای است.

۳-۲. الگوریتم رقابت استعماری دودویی

الگوریتم رقابت استعماری دودویی (BICA) مورد نظر، با اعمال تغییرات زیر روی الگوریتم رقابت استعماری پایه‌ای طراحی می گردد:

ایجاد کشورهای اولیه: یک مدل برنامه‌ریزی خطی صفر-یک با تعداد N_{var} متغیر تصمیم دارای یک جواب بهینه در فضای جستجوی $Feas=\{0,1\}^{N_{\text{var}}}$ می‌باشد. لذا در الگوریتم رقابت استعماری دودویی، هر کشور یک بردار $N_{\text{var}}-$ تایی خواهد بود که مولفه‌های آن اعداد 0 یا 1 می باشند. در گام آغازین، برای ایجاد کشورهای اولیه از تابع تولید جمعیت اولیه ارائه شده در [16] استفاده خواهیم کرد به طوری که این تابع به حداکثر $\left\lfloor \frac{N_{\text{var}}}{2} \right\rfloor$ متغیر تصمیم هرکشور اولیه مقدار 1 را تخصیص می‌دهد. بنابراین هر کشور تقریباً شبیه یک کروموزوم در الگوریتم ژنتیک عمل می‌نماید.

در الگوریتم BICA پیشنهادی، روند حرکت مستعمرات به سمت امپراطوری خود، عمدتاً شامل عملیات جذب (عملیات تلفیق) و عملیات جهش می باشد که در ذیل به تشریح آن‌ها می پردازیم.

عملیات جذب (تلفیق)**:** در الگوریتم BICA، عملگر جذب استفاده شده شبیه عملگر ترکیب متقاطع الگوریتم ژنتیک [17] بوده و در نتیجه روند عملیات جذب به صورت زیر خلاصه می گردد (شکل۲ را ملاحظه نمایید) که تعمیمی از رویکرد ارائه شده در [18] است:

رویکرد ۱: عملیات جذب

گام۱. شروع

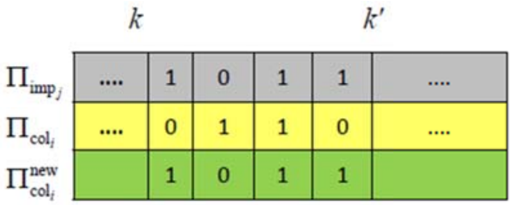
گام۲. قرار دهید $j=1$.

گام ۳. فرض کنید $\boldsymbol{I}_j$ مجموعه اندیس‌های مربوط به کشورهای مستعمره امپریالیست Π_{imp_j} باشد. به ازای هر $i\in \boldsymbol{I}_j$ عملیات زیر را انجام دهید:

(*): روی بردار امپریالیست
$\Pi_{\text{imp}_j}=\left[\pi_1^{\text{imp}_j},\pi_2^{\text{imp}_j},\ldots,\pi_{\text{N}_{\text{var}}}^{\text{imp}_j}\right]$
به طور تصادفی دو مؤلفه π_k و $\pi_{k'}$ را تعیین نمایید. به ازای هر $k\le s\le k'$، مقدار مؤلفه $\pi_s^{\text{imp}_j}$ امپریالیست Π_{imp_j} را جایگزین مؤلفه sام مستعمره Π_{col_i} تحت استعمار Π_{imp_j} نمایید. اگر مستعمره حاصل از این عملیات متعلق به فضای جواب مدل بهینه‌سازی تحت مطالعه نباشد، آنگاه به گام (*) برگردید.

گام۴. قرار دهید $j=j+1$. اگر $j\le N_{\text{imp}}$، آنگاه به گام ۳ برگردید.

گام۵. پایان.



شکل۲. عملیات جذب روی مستعمره Π_{col_i} تحت استعمار امپریالیست Π_{imp_j}

پس از انجام عملیات جذب، همانند الگوریتم ژنتیک یا الگوریتم استراتژی تکامل باید عملیات جهش نیز در هر تکرار از الگوریتم BICA انجام شود.

عملیات جهش: بعد از انجام عملیات جذب روی کشورهای مستعمره، عملیات جهش روی این کشورها انجام می‌پذیرد. یک روش برای عملیات جهش به‌شرح زیر می‌باشد [17]:

فرض کنید Π_{col_i} نشان‌دهنده یک مستعمره بعد از انجام عملیات جذب باشد. ابتدا یک بردار تصادفی
$\text{Rand}=(\text{rand}_1,\ldots,\text{rand}_{N_{\text{var}}})$
را تولید می کنیم. پارامتر P_{mutation} را به عنوان احتمال انجام جهش تعریف کرده و به آن مقداری متعلق به بازه $[0,1]$ تخصیص می‌دهیم. به ازای هر $1\le s\le N_{\text{var}}$، اگر
$\text{rand}_s<P_{\text{mutation}}$،
آنگاه یک عدد تصادفی باینری تولید کرده و آن را جایگزین مقدار فعلی مؤلفه sام مستعمره Π_{col_i} می‌کنیم.

در این مقاله، یک رویکرد دیگر نیز برای انجام عملیات جهش پیشنهاد می‌کنیم که آزمایش‌های محاسباتی ما نشان می‌دهند که این رویکرد بسیار کارا می باشد.

رویکرد ۲: عملیات جهش

گام۱. فرض کنید $\Pi_{\text{col}_i}=\left[\pi_1^{\text{col}_i},\pi_2^{\text{col}_i},\ldots,\pi_{\text{N}_{\text{var}}}^{\text{col}_i}\right]$ نشان دهنده یک مستعمره جدید بعد از انجام عملیات جذب باشد. یک بردار تصادفی
$\text{Rand}=(\text{rand}_1,\ldots,\text{rand}_{N_{\text{var}}})$

را تولید کنید به طوری که به ازای هر $1 \le s \le N_{\text{var}}$، مؤلفه rand_s یک عدد تصادفی گاوسی در بازه $[-0.5,\ 0.5]$ می باشد.

گام۲. به ازای هر $1 \le s \le N_{\text{var}}$، اگر

$$\pi_s^{\text{col}_i} + \text{Rand}_s > 1,$$

آنگاه قرار دهید $\pi_s^{\text{col}_i} = 1$. اگر

$$\pi_s^{\text{col}_i} + \text{Rand}_s < 0$$

آنگاه قرار دهید $\pi_s^{\text{col}_i} = 0$. در غیر این صورت قرار دهید

$$\pi_s^{\text{col}_i} = \begin{cases} 1 & ;\ \pi_s^{\text{col}_i} + \text{Rand}_s \ge 0.5 \\ 0 & ;\ \text{else} \end{cases}$$
.

اگر مستعمره حاصل در فضای جواب مدل بهینه‌سازی تحت مطالعه قرار نگیرد، آنگاه به گام ۱ برگردید.

با توجه به مباحث ارائه شده در بالا، الگوریتم رقابت استعماری دودویی به صورت زیر خلاصه می‌گردد:

الگوریتم BICA:

گام ۱. شروع

گام ۲. مطابق بخش۳-۲، با در نظر گرفتن محدودیت‌های مدل بهینه‌سازی تحت مطالعه، تعداد N_{country} کشور اولیه را ساخته و تعداد N_{imp} کشور امپریالیست اولیه را تشکیل دهید.

گام ۳. رویکرد ۱ را به منظور انجام عملیات جذب روی کشورهای مستعمره Π_{col_i} فراخوانی کنید.

گام ۴. عملیات جهش را با فراخوانی رویکرد ۲ (یا روش اول ارائه شده) روی کشورهای مستعمره انجام دهید.

گام ۵. اگر مستعمره‌ای در یک امپراطوری وجود داشته باشد که قدرت آن بیشتر از امپریالیست باشد، جای مستعمره و امپریالیست مربوطه را با هم عوض کنید.

گام ۶. با استفاده از رابطه (۷) هزینه کل هر امپراطوری را به دست آورید.

گام ۷. یک مستعمره از ضعیف‌ترین امپراطوری را انتخاب کرده و با بکارگیری بردار تخصیص (۱۰)، آن را به امپراطوری که بیش‌ترین احتمال استعمار را دارد، تخصیص دهید.

گام ۸. امپراطوری‌های ضعیف را حذف کنید.

گام ۹. اگر تنها یک امپراطوری باقی مانده باشد، توقف کنید، در غیر این صورت به گام ۳ برگردید.

گام ۱۰. پایان.

در بخش بعدی، به ارائه نتایج محاسباتی می‌پردازیم.

۴. نتایج محاسباتی

به منظور انجام آزمایش‌های محاسباتی، کد کامپیوتری الگوریتم BICA پیشنهادی را در محیط نرم‌افزار MATLAB نسخه 7.8.0.347 (R2009a) نوشته و روی یک کامپیوتر شخصی با پردازشگر Intel(R)core(TM)i5CPU1GHZ و حافظه 4GB فراخوانی کردیم. ما الگوریتم BICA و برنامه bintprog جعبه‌ابزار Optimization نرم‌افزار MATLAB را روی نمونه‌های مختلفی از مدل‌های مکان‌یابی $p-$میانه نیمه ناخوشایند با مشتریان به شکل زیرگراف اجرا نموده و در تمامی آزمایش‌های انجام گرفته، مشاهده کردیم که الگوریتم BICA دقیقاً به جواب‌های حاصل از برنامه bintprog نرم‌افزار MATLAB، به عبارتی به جواب‌های بهینه مدل‌های مکان‌یابی مورد آزمایش، همگرا گردیده است. در ضمن مشاهده گردید که زمان اجرای T_{BICA} الگوریتم BICA و زمان اجرای T_{MATLAB} برنامه MATLAB در رابطه

$$\frac{T_{BICA}}{T_{MATLAB}} \le \frac{66}{100}$$

صدق می‌کند. درضمن برای نمونه‌های با $N_{\text{var}} \ge 250$ ملاحظه کردیم که

$$\frac{T_{BICA}}{T_{MATLAB}} \le \frac{27}{100}$$

و در نتیجه نسبت $(T_{MATLAB} - T_{BICA}) / T_{BICA}$ روی مدل‌های با مقیاس بزرگ یک مقدار قابل توجهی می‌باشد. در جداول ۱و۲، میانگین زمان‌های اجرای ثبت شده برای نمونه های تست شده به ازای $p=2,\ p=4$ و ابعاد مختلف N_{var} ارائه شده‌اند.

جدول ۱. میانگین زمان‌های اجرای الگوریتم BICA و برنامه MATLAB به ازای $p=2$ و مقادیر مختلف N_{var}

N_{var}	MATLAB CPU (sec.)	BICA CPU (sec.)
30	0.7216	0.3112
40	1.6164	1.0284
60	2.9248	1.8987
80	8.2990	3.9737
100	10.5126	4.0000
130	11.0978	4.3213
180	13.0811	5.1101
210	14.2111	5.2111
240	17.3100	6.0010
270	25.0001	6.2421
300	32.0250	7.2212
330	39.9902	8.3560

جدول ۲. میانگین زمان‌های اجرای الگوریتم BICA و برنامه MATLAB به ازای $p=4$ و مقادیر مختلف N_{var}

N_{var}	MATLAB CPU (sec.)	BICA CPU (sec.)
30	1.0100	0.5500
40	1.9989	1.3200
60	3.2111	1.9200
80	8.1600	2.9372
100	14.4155	4.2166
130	15.2597	5.1142
180	18.0211	6.1821
210	20.2428	6.2321
240	26.4840	8.9112
270	33.5121	9.0000
300	40.2122	9.0100
330	50.2100	9.3550

با توجه به نتایج و مباحث بالا می‌توان اظهار کرد که در تمامی آزمایشات انجام گرفته الگوریتم BICA سریع‌تر از برنامه bintprog عمل نموده است. در ادامه این بخش، از بین تمامی نمونه‌های مکان‌یابی آزمایش شده، نتایج محاسباتی حاصل از اجرای الگوریتم BICA و برنامه نرم‌افزار MATLAB روی یک مثال به ازای اندازه‌های جمعیت ۵، ۱۵، ۲۵، ۵۰ و ۷۵ و کران بالای تکرار $\text{Maxit}=200$ ارائه گردیده و مورد بحث قرار گرفته است. در این مثال، شبکه G ارائه شده در شکل ۳ با مشتریان G_1, G_2, G_3, G_4 در نظر گرفته می‌شود به طوری که برای این مشتریان، وزن‌های حقیقی مقدار

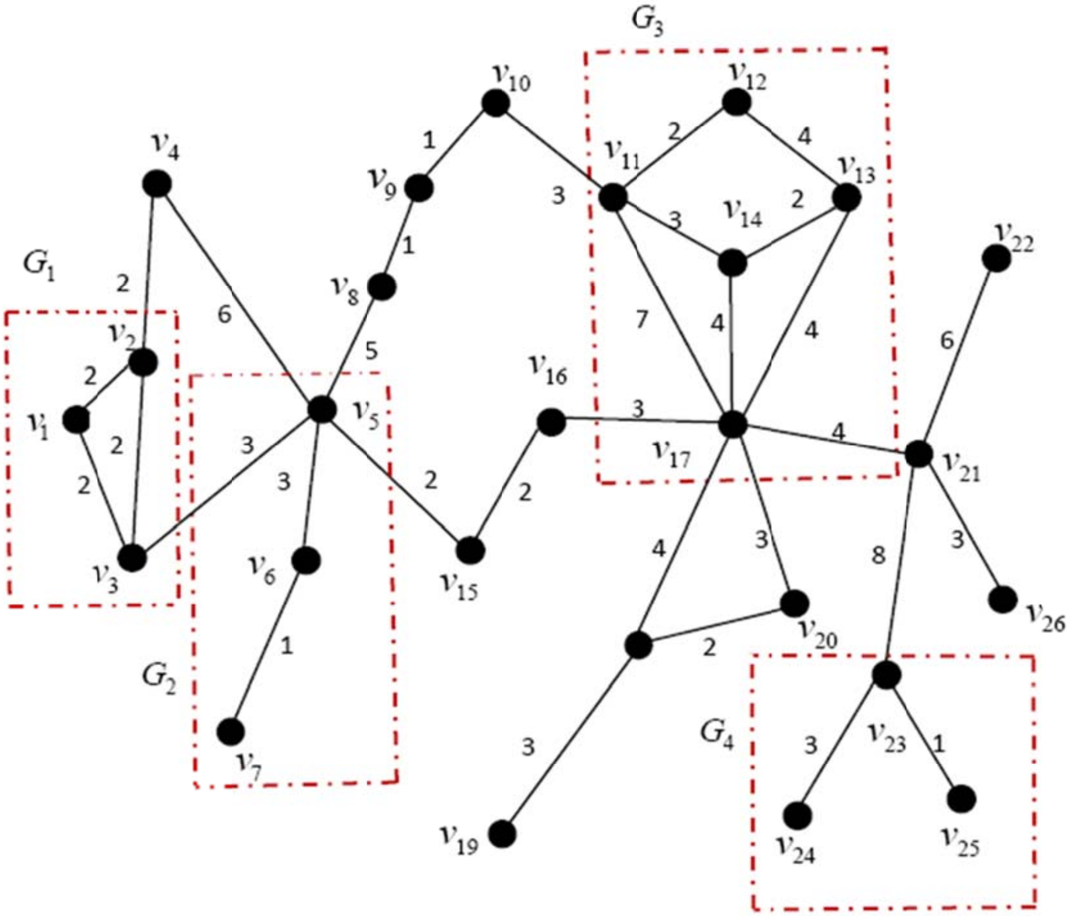
$$w_1=-4, \quad w_2=2, \quad w_3=-3, \quad w_4=5$$

و برای تمامی یال‌های شبکه G، طول‌های مثبت مطابق شکل ۳ تخصیص داده می‌شوند. هدف ما این است که یک مکان ۲-میانه نیمه ناخوشایند روی شبکه G داده شده پیدا کنیم. نتایج محاسباتی حاصل از اجرای الگوریتم BICA، روی مدل برنامه‌ریزی خطی صفر-یک (BLP) معادل، در جدول ۳ ارائه شده است. لازم به‌ذکر است که با توجه به زیاد بودن تعداد متغیرها و محدودیت‌های مدل BLP، از ارائه آن دراینجا صرف‌نظر شده است. در جدول ۳، نماد OI بیانگر کوچک‌ترین شماره تکراری می‌باشد که در پایان آن تکرار، الگوریتم BICA دقیقاً به یک جواب بهینه مسئله همگرا می‌گردد. نماد .PUP نشان‌دهنده اندازه جمعیت اولیه معین شده توسط تصمیم‌گیرنده برای الگوریتم می‌باشد.

جدول ۳. نتایج حاصل از اجرای الگوریتم BICA روی مثال داده شده با جمعیت‌های اولیه مختلف.

PUP.	5	15	25	50	75
Γ^*	-76.00	-76.00	-76.00	-76.00	-76.00
OI	81	72	65	60	54
CPU (sec.	3.002 0	3.192 0	3.909 0	4.200 0	5.000 2

از طرفی با اجرای برنامه bintprog نرم‌افزار MATLAB روی مدل BLP معادل مثال داده شده، مقدار هدف بهینه 76- با زمان اجرای (زمان پردازش) 12.4484 ثانیه به دست آمد.



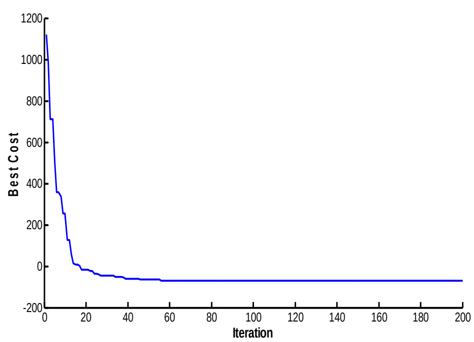
شکل۳. شبکه G با مشتریان G_1, G_2, G_3, G_4 و طول‌های یالی تخصیص یافته

از روی جدول ۳ می توان مشاهده کرد که بـه ازای انـدازه‌هـای مختلف جمعیت اولیه، الگوریتم BICA نیز توانسته است دقیقاً به مقدار بهینـه $\Gamma^* = -76$ مثـال داده شـده همگـرا گـردد. در ضمن به ازای اندازه‌هـای بـزرگ جمعیـت، الگـوریتم بـا تعـداد تکرارهای کمتری به جواب بهینه همگـرا شـده اسـت. نمـودار مقدار تابع هدف حاصل از اجرای الگوریتم BICA در شـکل ۴ نمایش داده شده است. جواب حاصل از اجرای الگوریتم BICA برای مدل BLP معادل، شامل

$$x_{1,25} = 1, \quad x_{2,25} = 1, \quad x_{3,25} = 1, \quad x_{4,24} = 1,$$

$$y_{24} = 1, \quad y_{25} = 1$$

می باشد که این نتیجه می‌دهد که راس‌های v_{24} ، v_{25} یک مکان ۲-میانه نیمه ناخوشایند روی شبکه شکل ۳ هستند.



شکل۴. نمودار مقدار تابع هدف مثال داده شده، در تکرارهای مختلف اجرای الگوریتم BICA

۵. نتیجه‌گیری و پیشنهادات

در این مقاله، یک الگوریتم رقابت استعماری دودویی (BICA) برای حل مدل‌های مکان‌یابی $p-$میانه نیمه ناخوشایند ($p-$میانه با وزن‌های راسی مثبت/منفی) روی شبکه‌ها با مشتریان به شکل زیرگراف پیشنهاد گردید. الگوریتم BICA را روی نمونه-

های مختلف اجرا کرده و از طریق مقایسه با جواب‌های بهینه حاصل از فراخوانی برنامه bintprog نرم افزار MATLAB مشاهده گردید که این الگوریتم نیز دقیقاً به جواب‌های نمونه‌های مورد آزمایش همگرا گردیده است. به علاوه در تمامی ازمایش-های انجام گرفته ملاحظه شد که الگوریتم BICA سریع‌تر از برنامه MATLAB عمل می کند. بنابراین می‌توان نتیجه گرفت که الگوریتم BICA قابل اطمینان بوده و از دقت و سرعت بالایی در حل مدل‌های مکان‌یابی (۲) برخوردار می‌باشد. البته همانند سایر الگوریتم‌های فراابتکاری، یک عیب الگوریتم BICA این است که به‌علت داشتن ساختار تصادفی ممکن است لازم باشد که در برخی موارد به منظور رسیدن به جواب‌های با دقت مطلوب، اجرای الگوریتم روی مسئله تحت مطالعه چند مرتبه تکرار شود.

برای انجام تحقیقات آتی، مطالعه و طراحی الگوریتم‌های فرا-ابتکاری دیگر با دقت و سرعت بالا جهت حل مسائل مکان‌یابی NP-سخت مختلف پیشنهاد می‌گردد.

پی نوشت

1- Vertex optimality property
2- NP-hard
3- Subgraph shaped customers
1. Median location problem
2. Semi obnoxious p-median location problem
3. Vertex optimality property
4. NP-hard
5. Subgraph shaped customers
6. Binary Linear Programming
7. Binary Imperialist Competitive Algorithm

مراجع

[۱] سیدحسینی، سیدمحمد، حیدری، روح‌اله، حیدری، طاهره، حل مساله مکان‌یابی پایانه‌های شبکه اتوبوس‌رانی درون‌شهری با استفاده از الگوریتم ژنتیک، نشریه بین‌المللی مهندسی صنایع و مدیریت تولید، (۱۳۸۸)، جلد ۲۰، شماره ۳ ، صفحه ۸۶-۷۵.

[۲] جبل‌عاملی، محمدسعید، شهانقی، کامران، نصیری، محمدرضا، ارائه مدل ترکیبی مکان‌یابی تسهیلات حساس، نشریه بین‌المللی مهندسی صنایع و مدیریت تولید، (۱۳۸۸)، جلد ۲۰، شماره ۴، صفحه ۷۶-۶۵.

[3] Francis R. L., McGinnis L. F., & White J. A., Facility Layout and Location, An Analytical Approach, Prentice Hall, Englewood Cliffs, (1992).

[4] Mirchandani P., & Francis R., Discrete Location Theory, Wiley, New York (1990).

[5] Hakimi S. L., Optimum location of switching centers and the absolute centers and medians of a graph, Operations Research, (1964), Vol. 12, pp. 450- 459.

[6] Kariv O., & Hakimi L., An algorithmic approach to network location problems, SIAM Journal of Applied Mathematics, (1979), Vol. 37, pp. 539- 560.

[7] Burkard R. E., Cela E., & Dollani H., 2-median in trees with pos/neg weights, Discrete Applied Mathematics, (2000), Vol. 105, pp. 51-71.

[8] Puerto J., Tamir A., Mesa J. A., & Prez-Brito D., Center location problems on tree graphs with subtree-shaped customers, Discrete Applied Mathematics, (2008), Vol. 156, pp. 2890-2910

[9] Cheng Y. K., Kang L.Y., & Lu C.H., The pos/neg-weighted 1-median problem on tree graphs with subtree-shaped customers, Theoretical Computer Science, (2010), Vol. 411, pp. 1038-1044.

[10] Zhang X., Kang L., & Cheng Y., The pos/neg weighted 1-median problem on block graphs with subgraph-shaped customers, Computing, Vol. 88, (2010), pp. 97-110

[11] Atashpaz-Gargari E., & Lucas C., Imperialist competitive algorithm: An algorithm for optimization inspired by imperialist competition, IEEE Congress on Evolutionary Computation, (2007), pp. 4661- 4667

[۱۲] حسینی نسب، حسن، براک، ساسان. حسینی، سیدمیثم. حل مسئله طراحی چیدمان FMS با سیستم حمل و نقل AGV بر اساس پایگاه دانش فازی و استفاده از الگوریتم رقابت استعماری و ژنتیک، نشریه بین‌المللی

مهندسی صنایع و مدیریت تولید، (۱۳۹۲)، جلد ۲۴، شماره ۱، صفحه ۷۹-۶۸.

[13] Mohammadi M., Tavakkoli-Moghaddam R., & Rostami H., A multi-objective imperialist competitive algorithm for a capacitated hub covering location problem, International Journal of Industrial Engineering Computations, , (2011), Vol. 2, pp. 671-688.

[14] Ardalan Z., Karimi S., Poursabzi O., & Naderi B., A novel imperialist competitive algorithm for generalized traveling salesman problems, Applied Soft Computing, (2015), Vol. 26, pp. 546-555.

[15] Hosseini S., & Al-Khaled A., A survey on the imperialist competitive algorithm meta-heuristic: Implementation in engineering domain and directions for future research, Applied Soft Computing, (2014), Vol. 24, pp. 1078-1094.

[16] Moadi S., Shariat Mohaymany A., & Babaie M., Application of imperialist competitive algorithm to the emergency medical services location problem, International Journal of Artificial Intelligence & Applications, (2011), Vol. 4, pp. 137-147.

[17] Holland J. H., Adaptation in Natural and Artificial Systems: *An Introductory* Analysis with Applications to Biology, Control, and Artificial Intelligence, MIT Press, (1992).

[18] Nozarian S., Soltanpoor H., & Vafaei-Jahan M., A binary model on the basis of imperialist competitive algorithm in order to solve the problem of knapsack 0-1, Proceedings of International Conference on System Engineering and Modeling, (2012).