حل برنامه‌ریزی دوسطحی[1] با SA[2]

سیدرضا حجازی*، عزیزاله معماریانی** و غلامرضا جهانشاهلو***

بخش مهندسی صنایع دانشکدهٔ فنی و مهندسی، دانشگاه تربیت مدرس

بخش ریاضی، دانشگاه تربیت معلم

(دریافت مقاله: ۷۸/۱۱/۱۸ - دریافت نسخهٔ‌نهایی: ۷۹/۸/۱)

چکیده - برنامه ریزی دو سطحی ابزاری برای مدلسازی مسئلهٔ تصمیم‌گیری غیرمتمرکز است که در آن تصمیم گیرندهٔ سطح یک و دو به ترتیب رهبر و پیرو گفته می‌شوند. ثابت شده است که مسئلهٔ برنامه ریزی دوسطحی یک مسئلهٔ Np-har است. روشهای زیادی برای حل این مسئله ارائه شده است؛ اما کارایی محاسباتی[3] آنها درحدی نیست که بتوانند مسائل بزرگ را حل کنند. در این مقاله سعی شده است روشی براساس SA برای حل مسئلهٔ برنامه ریزی دوسطحی توسعه داده شود. این روش از روشهای مدرن ابتکاری[4] است که می‌تواند مسائل بزرگ را نیز تا رسیدن به یک جواب نزدیک به جواب بهینه حل کند.

در این مقاله همچنین با حل مسائل متعدد روش پیشنهادی با روش ارائه شده توسط "ماتیو" و همکارانش [۱] مقایسه شده است.

Simulated Annealing Approach for Solving Bilevel Programming Problem

S. R. Hejazi, A. Memariani and G. R. Jahanshahloo

Department of Industrial Engineering, School of Engineering, Tarbiat Modarres University

Department of Mathematics, Teacher Training University

ABSTRACT- ***Bilevel programming, a tool for modeling decentralized decision problems, consists of the objective of the leader at its first level and that of the follower at the second level. Bilevel programming has been proved to be an Np-hard problem. Numerous algorithms have been developed for solving bilevel***

---

*- دانشجوی دکترا　　**- استادیار　　***- استاد

programming problems. These algorithms lack the required efficiency for solving a real problem. In this paper, attempts have been made to develop an algorithm based on Simulated Annealing (SA) Approach. This algorithm is efficient enough to find a near optimal solution. The proposed method is compared with the one developed by Mathieu et al [1] through application of both to a number of different problems.

فهرست علائم

A_1	ماتریس ضرایب x در محدودیتهای مدل	m	تعداد محدودیتهای مدل	u	بردار متغیرهای کمبود[5] محدودیتها
A_2	ماتریس ضرایب y در محدودیتهای مدل	n_0	تعداد تکرارها در هر دما	v	بردار متغیرهای مازاد[6] برای همزاد[7] مسئلهٔ سطح دوم
b	بردار منابع مدل	n_1	تعداد مؤلفه‌های بردار x	x	بردار متغیرهای تحت کنترل سطح یک
c_1	بردار ضرایب x در تابع هدف سطح یک	n_2	تعداد مؤلفه‌های بردار y	y	بردار متغیرهای تحت کنترل سطح دو
c_2	بردار ضرایب x در تابع هدف سطح دو	R	احتمال جستجوی همسایگی جوابهای نشدنی، که تابعی از دمای الگوریتم است	z	بردار متغیرهای همزاد مسئلهٔ سطح دوم
d_1	بردار ضرایب y در تابع هدف سطح یک	r	عدد تصادفی با توزیع یکنواخت، در فاصلهٔ [۰،۱]	Δ	درصد انحراف از مقدار بهینه
d_2	بردار ضرایب y در تابع هدف سطح دو	T	دمای الگوریتم در مراحل مختلف	Δ_{GA}	درصد انحراف از مقدار بهینه برای الگوریتم GA
F	مقدار تابع هدف سطح یک، به دست آمده از الگوریتم	T_0	دمای اولیهٔ الگوریتم	Δ_{SA}	درصد انحراف از مقدار بهینه برای الگوریتم SA
F(x,y)	تابع هدف تصمیم گیرندهٔ سطح یک	t_{GA}	زمان رسیدن به جواب با الگوریتم GA	ε	عدد حقیقی بسیار کوچک
F°	مقدار بهینهٔ تابع هدف سطح یک				
f(x,y)	تابع هدف تصمیم گیرندهٔ سطح دو	t_{SA}	زمان رسیدن به جواب با الگوریتم SA پیشنهادی		
f_0	ضریب کاهش دما				

۱- مقدمه

بیشتر مدلهای ریاضی شامل یک تصمیم گیرنده و یک تابع هدف هستند که برای برنامه ریزی متمرکز به کار می‌روند، اما برنامه‌ریزی ریاضی دو سطحی برای تصمیم‌گیری غیرمتمرکز توسعه داده شده است. در برنامه ریزی دوسطحی (BLP) که تصمیم گیرندهٔ سطح یک آن را رهبر و سطح دو آن را پیرو می‌گوییم، هر تصمیم گیرنده سعی می‌کند تابع هدف خود را بدون توجه به هدف قسمت دیگر بهینه کند؛ اما تصمیم هر تصمیم گیرنده بر مقدار تابع هدف و فضای تصمیم‌گیری سطح دیگر اثر می‌گذارد. شکل ریاضی برنامه ریزی دوسطحی به صورت زیر است:

$$\max_x F(x,y) = c_1x + d_1y$$

$$\max_y f(x,y) = c_2x + d_2y \qquad (۱)$$

$$\text{s.t: } \begin{cases} A_1x + A_2y \le b \\ x,y \ge 0 \end{cases}$$

نقطهٔ (x_0,y_0) را یک نقطهٔ شدنی می‌گوییم اگر در محدودیتهای مدل (۱) صدق کند و آن را یک نقطهٔ قابل دستیابی می‌گوییم اگر برای $x=x_0$، y_0 جواب بهینهٔ مسئلهٔ سطح دوم باشد. مسئلهٔ سطح دوم به صورت زیر تعریف می‌شود.

$$\max \quad d_2y + c_2x_0 \qquad (۲)$$

$$\begin{cases} A_2y \le b - A_1x_0 \\ y \ge o \end{cases}$$

که در آن x_0 توسط سطح یک تعیین شده است. همچنین مجموعهٔ تمام نقاط قابل دستیابی را ناحیه قابل دستیابی[8] مسئلهٔ BLP می‌گوییم که زیرمجموعه‌ای از مجموعه جوابهای شدنی BLP است و لزوماً یک مجموعهٔ محدب نیست. هدف از حل مسئلهٔ BLP به دست آوردن نقطه‌ای از فضای قابل دستیابی است که به ازای آن مقدار تابع هدف سطح یک بر روی ناحیه قابل دستیابی بهینه شده باشد. از ویژگیهای مهم ناحیه قابل دستیابی این است که، مجموعهٔ

نقاط رأسی[9] آن، زیرمجموعهٔ نقاط رأسی فضای شدنی است و جواب بهینهٔ BLP نیز یکی از این نقاط رأسی است.

روشهای ارائه شده برای حل این مسئله را می‌توان به صورت زیر دسته بندی کرد:

الف - روشهایی براساس شمارش رئوس[10]

ب - روشهایی براساس شرایط کوهن - تاکر

ج - نگرش فازی[11] برای حل BLP

د - روشهای مدرن ابتکاری برای حل BLP

الف - روشهایی براساس شمارش رئوس

اساس این دسته از روشها بر این ایده استوار است که نقاط رأسی ناحیه قابل دستیابی برای مسئلهٔ BLP، نقطهٔ رأسی فضای شدنی مسئلهٔ مورد نظر نیز هستند، همچنین جواب بهینهٔ مسئله نیز یکی از این نقاط رأسی است.

اولین روش براین اساس را فالک [۲] برای مسئلهٔ Max-Min که یک مورد خاص از BLP است، توسعه داده است. در این روش نقاط رأسی مسئلهٔ سطح یک، با یک روش شاخه و کرانه[12] جستجو می‌شود. بیالاس و کاروان [۳] الگوریتمی تحت عنوان "Kامین بهترین"[13] ارائه کرده‌اند که یکی از روشهای مهم توسعه داده شده برای حل این مسئله است، در این روش فرایند جستجو از ناحیه غیرقابل دستیابی شروع شده و اولین نقطهٔ قابل دستیابی که پیدا شود، متوقف می‌شود و این نقطه جواب بهینهٔ مسئلهٔ BLP است.

بیالاس و کاروان [۴] همچنین روشی را پیشنهاد کرده‌اند که از یک نقطهٔ رأسی ناحیه قابل دستیابی جستجو را آغاز می‌کند، اما این روش ممکن است در یک جواب بهینهٔ موضعی[14] متوقف شود.

الگوریتم دیگری به نام GSA[15] توسط برد [۵] و به نام BCP[16] توسط انلو [۶] بر پایه برنامه ریزی خطی دومعیاری توسعه داده شده است، ولی این روش توسط کندلر [۷]، کلارک و وستربرگ [۸]، ون و سو [۹]، بن ایاد و بلایر [۱۰] و بالاخره توسط مارکوته و ساوارد [۱۱] با ارائهٔ مثالهای نقض مورد انتقاد قرار گرفته و نشان داده شده است که مفاهیم پارتو و بهینگی برنامه ریزی دوسطحی کاملاً متفاوت بوده و نمی‌توان آنها را به سادگی به هم ارتباط داد.

ب - روشهایی براساس شرایط کوهن - تاکر

در این دسته از روشها شرایط کوهن - تاکر برای تابع هدف سطح دوم جایگزین شده و مسئله با تابع هدف سطح یک حل می‌شود. آمات و مک کارل [۱۲] برای حل این مسئله محدودیتهای مکمل[17] به وجود آمده را به محدودیتهای خطی با متغیرهای صفر و یک تبدیل کرده و مسئله را حل می‌کنند. بیالاس و کاروان [۳ و ۴] الگوریتم چرخش لولایی مکمل پارامتری[18] (PCP) را توسعه داده‌اند، در این روش تابع هدف به صورت محدویتی به شکل، $c_1x+d_1y\leq\alpha$ به محدودیتهای مسئله اضافه شده و به یک مسئلهٔ مکمل خطی[19] (LCP) تبدیل می‌شود. اگر این مسئله شدنی باشد، با افزایش α می‌توان به یک جواب بهتر دست یافت. اگر در مرحله‌ای، LCP شدنی نباشد، جواب شدنی مرحلهٔ قبل جواب مسئلهٔ مورد نظر با این روش است.

برد و مور [۱۳] نیز روش شاخه و کرانه‌ای برای حل این مسئله توسعه داده‌اند. همچنین وایت و آناندالینگام [۱۴] و آناندالینگام و وایت [۱۵] با اضافه کردن یک تابع جریمه[20] به عنوان بخشی از تابع هدف، جهت ارضای محدودیتهای مکمل، مسئلهٔ مورد نظر را حل کرده‌اند.

ج - نگرش فازی برای حل BLP

با این نگرش در سال ۱۹۹۶ روشی توسط شیه و همکارانش [۱۶] ارائه شده است که پس از آن ساکاوا، نیشی زاکی و یومورا [۱۷] این روش را با اصطلاحاتی ارائه کرده‌اند. در این روش اصل عدم همکاری[21] که یکی از اصول BLP است حذف شده است و در واقع روشی برای رسیدن به یک جواب توافقی ارائه شده است.

د - روشهای مدرن ابتکاری برای حل BLP

در این زمینه در سال ۱۹۹۴ ماتیو، پیتارد و آناندالینگام [۱] روشی براساس الگوریتم ژنی[22] ارائه کرده‌اند، روش ارائه شده، تمام ناحیه قابل دستیابی را جستجو می‌کند و هر کروموزوم[23] شدنی آن معرف یک جواب قابل دستیابی BLP است.

ساهین و سیریت [۱۸] روشی براساس سیمیولیتد انیلینگ ارائه کرده‌اند. ایدهٔ اصلی این روش بسط فضای جستجو با فازی سازی مسئله و جستجوی تصادفی به کمک زنجیره‌های مارکوف[24]

است. آنها مسائل کم و کوچکی را حل و گزارش کرده‌اند که گزارش ارائه شده نشان می‌دهد این روش کارایی مناسبی برای حل برنامه ریزی دوسطحی خطی را ندارد. به عنوان نمونه مسئلهٔ اول گزارش شده که یک برنامه ریزی دوسطحی خطی است، دارای سه متغیر و پنج محدودیت است که الگوریتم آنها در حدود ۴۴ ثانیه آن را حل می‌کند. اما به هر حال این روش قادر است هر برنامه ریزی دوسطحی مانند خطی، غیرخطی یا عدد صحیح را حل کند.

روشی نیز براساس الگوریتم ژنی توسط حجازی و همکارانش [۱۹] توسعه داده شده است که در این روش هر کروموزوم شدنی معرف یک نقطهٔ رأسی ناحیه قابل دستیابی است و نسبت به روش ماتیو، پیتارد و آناندالینگام [۱] فضای بسیار محدودتری را جستجو می‌کند.

روشی که در این مقاله ارائه می‌شود بر اساس SA توسعه داده شده است و نقاط رأسی فضای قابل دستیابی را جستجو می‌کند.

۲- به کارگیری SA برای حل BLP

مدل برنامه ریزی دوسطح (۱) را در نظر می‌گیریم، در ابتدا شرایط کوهن - تاکر را برای مسئلهٔ سطح دوم نوشته و در مدل جایگزین می‌کنیم. در این صورت خواهیم داشت:

$$\text{Max } c_1x + d_1y$$

$$\text{s.t:} \begin{cases} A_1x + A_2y + u = b \\ zA_2^T - v = d_2 \\ uz = 0 \qquad\qquad vy = 0 \\ x,y,z,u,v \geq 0 \end{cases} \qquad (۳)$$

مدل (۳) یک برنامه ریزی ریاضی غیرخطی است. اگر u، z، v و yها را به گونه‌ای محدود کنیم که محدودیتهای $uz=0$ و $vy=0$ برقرار باشد، می‌توانیم این محدودیتها را حذف کرده و برنامه ریزی خطی زیر را داشته باشیم، که فضای جوابهای شدنی آن بخشی از فضای جوابهای شدنی مدل (۳) است.

$$\text{Max } c_1x + d_1y'$$

$$\text{s.t:} \begin{cases} A_1x + A_2y' + u' = b \\ z'A_2^T - v' = d_2 \\ x,y',z',u',v' \geq 0 \end{cases} \qquad (۴)$$

برای ساختن مدل (۴) با ویژگی ذکر شده (هر جواب شدنی مدل (۴) یک جواب شدنی مدل (۳) است)، از یک رشتهٔ اعداد به طول $m+n_2$ که هر عدد (جزء) آن می‌تواند صفر یا یک باشد، به عنوان راهنمای تغییرات استفاده می‌کنیم که m تعداد محدودیتهای مدل (۱) و n_2 تعداد متغیرهای تحت کنترل تصمیم گیرندهٔ سطح دو یا اندازهٔ بردار y است.

نحوهٔ تغییر مدل (۳) با استفاده از رشتهٔ اعداد تعریف شده به این صورت است که، m جزء اول این رشته را متناظر m معادلهٔ $uz=0$ و n_2 جزء آخر آن را متناظر n_2 معادلهٔ $vy=0$ در نظر می‌گیریم. اگر جزء iام از m جزء اول عدد یک است، در مدل (۴) u_i را مساوی صفر قرار می‌دهیم. اگر جزء iام از m جزء اول عدد صفر است، در مدل (۴) z_i را مساوی صفر قرار می‌دهیم تا همواره u_iz_i در مدل (۴) مساوی صفر باشد. همچنین اگر جزء jام از بین n_2 جزء آخر رشتهٔ اعداد مورد نظر عدد یک است، در مدل (۴) y_j را مساوی صفر قرار می‌دهیم و اگر این جزء عدد صفر است، در مدل (۴) v_j را مساوی صفر قرار می‌دهیم در این صورت در مدل (۴)، v_jy_j همواره مساوی صفر خواهد بود. بدین ترتیب مدل (۴) به گونه‌ای ساخته می‌شود که مجموعه جوابهای شدنی آن زیرمجموعه‌ای از مجموعه جوابهای شدنی مدل (۳) باشد.

تعریف ۱- یک رشته اعداد به طول $m+n_2$ که هر جزء آن عدد صفر یا یک باشد را یک جواب الگوریتم SA می‌نامیم.

تعریف ۲- اگر مدل (۴)، متناظر یک جواب SA شدنی باشد، این جواب را شدنی و در غیر این صورت آن را نشدنی می‌گوییم.

با توجه به تعاریف بالا، فضای جوابهای SA برای مدل (۳) گسسته و تعداد جوابهای آن 2^{m+n_2} است، که برخی از آنها شدنی و برخی نشدنی هستند.

قضیه ۱- فرض کنیم مجموعه جوابهای مدل (۳)، S باشد و N تعداد جوابهای شدنی SA باشد که مجموعه جوابهای شدنی مدل (۴)، مربوط به هر کدام را با $B_1,B_2,B_3,...,B_N$ نشان دهیم، آن گاه داریم

$$S = \bigcup_{i=1}^{N} B_i$$

اثبات - بدیهی است که هر جواب متعلق به B_i، $(i=1,2,...,N)$،

[DOR: 20.1001.1.22287698.1379.19.2.6.6]

چون تمام محدودیتهای مدل (۳) را ارضا می‌کند متعلق به S نیز است، یعنی داریم:

$$\bigcup_{i=1}^{N} B_i \subseteq S \qquad \text{(الف)}$$

از طرف دیگر اگر جوابی مانند (x_0,y_0,u_0,v_0,z_0) متعلق به S باشد؛ آن گاه به ازای $(i=1,2,...,m)$ و $u_{0i}z_{0i}=0$ ؛ باید حتماً u_{0i} و یا z_{0i} مساوی صفر باشد. در این صورت جواب SA مانند P را می‌سازیم؛ به طوری که اگر u_{0i} مساوی صفر است، جزء iام از m جزء اول آن عدد یک باشد، اگر u_{0i} بزرگتر از صفر است، جزء iام مساوی صفر باشد و اگر u_{0i} و z_{0i} هر دو مساوی صفر باشند، آن گاه این جزء می‌تواند به طور اختیاری عدد صفر یا یک انتخاب شود.

همچنین به ازای $v_{0j}y_{0j}=0$ و $(j=1,2,...,n_2)$، حتماً باید v_{0j} و یا y_{0j} مساوی صفر باشد. اگر y_{0j} مساوی صفر است، در P، جزء jام از n_2 جزء آخر را برابر عدد یک قرار می‌دهیم و اگر y_{0j} بزرگتر از صفر است، این جزء را صفر قرار می‌دهیم و در صورتی که v_{0j} و y_{0j} هر دو صفر باشند، جزء مورد نظر را به طور اختیاری عدد صفر یا یک انتخاب می‌کنیم.

بدین ترتیب P یک جواب شدنی SA خواهد بود که (x_0,y_0,u_0,v_0,z_0)، یک جواب شدنی مدل (۴) متناظر آن است. در نتیجه می‌توان نوشت:

$$S \subseteq \bigcup_{i=1}^{N} B_i \qquad \text{(ب)}$$

از روابط (الف) و (ب) می‌توان نتیجه گرفت که؛

$$S = \bigcup_{i=1}^{N} B_i$$

۲-۱- روش حل مدل (۴)

برای حل مدل (۴) متناظر یک جواب SA، می‌توان ابتدا این مدل را به دو مسئلهٔ کاملاً از هم جدا به صورت زیر تجزیه کرد:

$$\text{Max } c_1x + d_1y'$$

$$\text{s.t}: \begin{cases} A_1x + A_2y' + u' = b \\ x,y',u' \geq 0 \end{cases} \qquad (۵)$$

$$\begin{cases} z'A_2^T - v' = d_2 \\ z',v' \geq 0 \end{cases} \qquad (۶)$$

چون دو مسئلهٔ (۵) و (۶) هیچ متغیر مشترکی ندارند تجزیه بالا مجاز است. حال اگر هر دو مسئلهٔ بالا شدنی باشند، مدل (۴) شدنی است در غیر این صورت مدل (۴) نشدنی است.

۲-۲- بررسی شدنی بودن یک جواب SA

برای اینکه تعیین کنیم یک جواب SA مانند P شدنی است یا نه، ابتدا مسئلهٔ (۵) متناظر آن را می‌سازیم و حل می‌کنیم، اگر نشدنی باشد آن گاه P نیز نشدنی خواهد بود. در غیر این صورت با توجه به جواب بهینهٔ مسئلهٔ (۵)، P را بازسازی می‌کنیم، یعنی اگر جزء iام P که متناظر y_i است صفر باشد و در جواب به دست آمده y_i مساوی صفر شده است، آن گاه جزء iام را به عدد یک تغییر می‌دهیم. همچنین اگر جزء jام P که متناظر u_j است، عدد صفر باشد و در جواب بهینهٔ مسئلهٔ (۵)، u_j نیز صفر شده باشد، آن گاه جزء jام P را عدد یک قرار می‌دهیم. اکنون مسئلهٔ (۶)، متناظر P بازسازی شده را حل می‌کنیم (مسئلهٔ (۶) متناظر با P بازسازی شده دارای فضای شدنی بزرگتر یا برابر فضای شدنی مسئلهٔ (۶)، متناظر با P قبل از بازسازی است). اگر این مسئله شدنی باشد، آن گاه P نیز یک جواب شدنی SA خواهد بود و در غیر این صورت P یک جواب نشدنی SA است.

قضیهٔ ۲- جواب بهینهٔ به دست آمده از مدل (۴) مربوط به یک جواب شدنی SA، متناظر یک نقطهٔ رأسی ناحیهٔ قابل دستیابی مدل (۱) است.

اثبات - فرض کنیم جواب بهینهٔ مدل (۴) $(x_0,y_0',u_0',v_0',z_0')$ باشد. چون این جواب یک جواب مدل (۳) است، در شرایط کوهن - تاکر برای مسئلهٔ سطح دوم نیز صدق می‌کند، پس یک جواب قابل دستیابی برای مدل (۱) خواهد بود. اما از طرفی (x_0,y_0',u_0') جواب بهینهٔ مدل (۵) است، در نتیجه یک جواب پایه برای این مسئله است. یعنی ستونهای A_1 و A_2 متناظر با مؤلفه‌های مخالف صفر x_0 و y_0' مستقل خطی هستند. پس (x_0,y_0') یک نقطهٔ رأسی $\{A_1x + A_2y \leq b : x,y \geq 0\}$ خواهد بود. از طرفی گفتیم این جواب یک جواب قابل دستیابی است، پس (x_0,y_0') یک نقطهٔ رأسی ناحیهٔ قابل دستیابی برای مدل (۱) است.

با توجه به قضیهٔ (۲)، الگوریتمی که توسعه داده می‌شود نقاط رأسی ناحیه قابل دستیابی را جستجو و ارزیابی می‌کند.

قضیه ۳- اگر مسئلهٔ (۶)، برای یک جواب SA مانند P نشدنی باشد؛ هر جواب دیگری مانند Q که اجزای صفر P در آن نیز صفر باشند، نشدنی خواهد بود.

اثبات - فرض کنیم مسئلهٔ (۶)، متناظر P به صورت زیر باشد:

$$\begin{cases} z' A_2^T - v' = d_2 \\ z', v' \geq 0 \end{cases} \qquad (۷)$$

در این مدل بعضی از z_iها و v_jها مساوی صفرند. اگر مسئلهٔ (۶) متناظر Q به صورت زیر باشد:

$$\begin{cases} z'' A_2^T - v'' = d_2 \\ z'', v'' \geq 0 \end{cases} \qquad (۸)$$

در این مدل علاوه بر z_i و v_iهایی که در (۷) مساوی صفرند، ممکن است z_i و v_jهای دیگری نیز صفر باشند، به همین دلیل فضای شدنی (۸) زیرمجموعه‌ای از فضای شدنی (۷) است، پس می‌توان نتیجه گرفت اگر (۷) نشدنی باشد، (۸) نیز نشدنی است، یعنی Q نیز یک جواب نشدنی SA خواهد بود.

قضیه ۴- اگر مسئلهٔ (۵)، برای یک جواب SA مانند P نشدنی باشد، هر جواب دیگری مانند Q که اجزای عدد یک در آن نیز عدد یک باشند، نیز نشدنی خواهد بود.

اثبات - فرض کنیم مسئلهٔ (۵) متناظر P به صورت زیر باشد:

$$\text{Max } c_1 x + d_1 y'$$

$$\text{s.t: } \begin{cases} A_1 x + A_2 y' + u' = b \\ x, y', u' \geq 0 \end{cases} \qquad (۹)$$

در مدل (۹) بعضی از مؤلفه‌های بردارهای y' و u' مساوی صفرند. اگر مسئلهٔ (۵) متناظر Q به صورت زیر باشد:

$$\text{Max } c_1 x + d_1 y''$$

$$\text{s.t: } \begin{cases} A_1 x + A_2 y'' + u'' = b \\ x, y'', u'' \geq 0 \end{cases} \qquad (۱۰)$$

آن گاه در این مدل علاوه بر u_i و y_jهایی که در مدل (۹) مساوی صفرند ممکن است u_i و y_iهای دیگر نیز مساوی صفر باشند، به همین دلیل فضای شدنی مدل (۱۰) زیرمجموعهٔ فضای شدنی مدل (۹) خواهد بود. بنابراین اگر مدل (۹) نشدنی باشد مدل (۱۰) نیز نشدنی است یعنی Q نیز یک جواب نشدنی SA است.

از قضایای (۳) و (۴) برای تشخیص جوابهای SA نشدنی قبل از حل مدلهای (۵) و (۶) و با استفاده از اطلاعات مراحل قبل در الگوریتم مورد نظر استفاده خواهد شد.

تعریف ۳- اگر P یک جواب SA باشد، Q را در همسایگی مرتبهٔ اول P می‌گوییم اگر P و Q تنها در یک جزء با هم مختلف باشند. به عبارت دیگر مجموعهٔ تمام جوابهای SA را که فقط در یک جزء با هم تفاوت دارند را همسایگی مرتبهٔ اول P می‌نامیم.

تعریف ۴- مجموعهٔ تمام جوابهای SA که دقیقاً در i جزء با P تفاوت دارند را همسایگی مرتبهٔ iام P می‌نامیم.

با توجه به تعاریف بالا، مرتبهٔ همسایگی حداکثر $m+n_2$ است.

۲-۳- نحوهٔ ساختن جواب اولیه الگوریتم SA

برای ساختن جواب اولیه الگوریتم ابتدا مدل شمارهٔ (۱۱) را حل می‌کنیم، سپس یک جواب SA را به این صورت می‌سازیم که، اگر u_i بزرگتر از صفر باشد، جزء iام از m جزء اول P را مساوی عدد یک قرار می‌دهیم، در غیر این صورت آن را برابر صفر قرار می‌دهیم. همچنین اگر y_j بزرگتر از صفر باشد، جزء jام از n_2 جزء آخر P را مساوی عدد یک قرار می‌دهیم، در غیر این صورت آن جزء را برابر صفر قرار می‌دهیم. بدین ترتیب P به عنوان یک جواب الگوریتم SA ساخته می‌شود.

$$\text{Max } c_2 x + d_2 y$$

$$\begin{cases} A_1 x + A_2 y + u = b \\ x, y \geq 0 \end{cases} \qquad (۱۱)$$

۳- گامهای الگوریتم SA برای حل BLP

گام صفر:

الف - $T \leftarrow T_0$، $f \leftarrow f_0$ و $n \leftarrow 0$ (T_0 دمای اولیه، f_0 ضریب کاهش دما و n یک شمارنده برای تعداد تکرارها در هر دماست).

ب - با روشی که در بخش (۲-۳) توضیح داده شده است، یک جواب شدنی اولیه مانند P برای ادامهٔ الگوریتم می‌سازیم. این جواب را با نام Q به عنوان بهترین جواب پیدا شده تا به حال، نگهداری می‌کنیم و به گام یک می‌رویم.

گام یک:

۱- یک جواب در همسایگی مرتبهٔ iام جواب فعلی (مرتبهٔ همسایگی با توجه به نحوهٔ استفاده از همسایگی با مراتب مختلف که در بخش (۴) مشخص خواهد شد، تعیین می‌شود). مانند P_1 را به طور تصادفی تولید می‌کنیم (برای تولید P_1، به طور تصادفی i جزء جواب فعلی را انتخاب کرده و هر کدام از اجزای انتخاب شده صفر باشد به عدد یک تغییر می‌دهیم و بالعکس).

۲- با استفاده از اطلاعات ذخیره شده از تکرارهای قبل و به کارگیری قضایای (۳) و (۴)

۲-۱- اگر مسئلهٔ (۵) متناظر P_1 شدنی است

۲-۱-۱ اگر مسئلهٔ (۶) متناظر P_1 نیز شدنی باشد به گام دو می‌رویم

۲-۱-۲ در غیر این صورت به گام سه می‌رویم.

۲-۲- اگر مسئلهٔ (۵) متناظر P_1 نشدنی باشد به گام سه می‌رویم.

۲-۳- اگر شدنی یا نشدنی بودن مسائل (۵) و (۶) متناظر P_1 تشخیص داده نشود. در این صورت مسئلهٔ (۵) متناظر P_1 را حل می‌کنیم

۲-۳-۱ اگر مسئله شدنی نباشد، اطلاعات مربوطه را در صورت نیاز ذخیره کرده و به گام سه می‌رویم.

۲-۳-۲ اگر مسئله شدنی باشد با توجه به جواب بهینهٔ مسئلهٔ (۵)، P_1 را بازسازی کرده و مسئلهٔ (۶)، متناظر P_1 بازسازی شده را حل می‌کنیم

۲-۳-۲-۱ اگر مسئله نشدنی باشد، اطلاعات مربوطه را در صورت نیاز ذخیره کرده و به گام سه می‌رویم

۲-۳-۲-۲ اگر مسئله شدنی باشد، اطلاعات مربوطه را در صورت نیاز ذخیره کرده و به گام دو می‌رویم.

گام دو: $P \leftarrow P_i$، $n \leftarrow n+1$

اگر جواب پیدا شده بهتر از Q باشد، Q را برابر این جواب قرار داده و به گام چهارم می‌رویم. در غیر این صورت، بدون تغییر Q به گام چهارم می‌رویم.

گام سه:

عدد تصادفی r را با توزیع یکنواخت در فاصلهٔ [۱ و ۰] تولید می‌کنیم.

- اگر $r \leq R$ باشد، آن‌گاه؛ $P \leftarrow P_1$، $n \leftarrow n+1$ و به گام چهارم می‌رویم

- اگر $r > R$ باشد، آن‌گاه؛ $n \leftarrow n+1$ و به گام چهارم می‌رویم.

گام چهارم:

اگر $n > n_0$؛ به گام یک می‌رویم.

- در غیر این صورت، $T \leftarrow f \times T$، $n \leftarrow 0$،

- اگر $T \leq \varepsilon$ است (یا الگوریتم به جواب مورد نظر رسیده است)، توقف می‌کنیم و Q را به عنوان جواب BLP مورد نظر با این الگوریتم معرفی می‌کنیم.

- در غیر این صورت به گام یک بر می‌گردیم.

۴- تجزیه و تحلیل نتایج محاسباتی

برای تجزیه و تحلیل نتایج محاسباتی الگوریتم توسعه داده شده، مسائل برنامه‌ریزی خطی را با نرم‌افزار LINDO حل می‌کنیم و یک برنامهٔ رابط به زبان C^{++} نوشته شده است که جوابهای به دست آمده از حل مسئله‌های خطی توسط LINDO را گرفته و طبق الگوریتم توسعه داده شده جواب BLP مورد نظر را جستجو می‌کند.

در این بخش ابتدا تابع R تعیین شده است و سپس مقدار مناسب پارامترهای آن با حل مسائل متعددی تخمین زده شده است. همچنین در ادامه پارامترهای دیگر الگوریتم مانند ضریب تغییر دما، تعداد تکرارها در هر دما و نحوهٔ استفاده از همسایگیها با مراتب مختلف مورد بررسی قرار گرفته است.

پس از تعیین پارامترهای مختلف الگوریتم در انتهای این بخش روش SA پیشنهادی و روش GA ارائه شده توسط ماتیو و همکارانش با حل مسائلی با اندازه‌های مختلف مقایسه شده است.

۴-۱- تعیین تابع R

به منظور تعیین تابع مناسب R ابتدا با حل مسائلی، چندین تابع مورد بررسی قرار گرفته است که توابع زیر برای بررسی نهایی انتخاب شده‌اند.

الف - $R_1 = 0.3$

ب - $R_2 = EXP(C \times T/T_0)$

ج - $R_3 = A/(EXP(B \times (T_0 - T))$

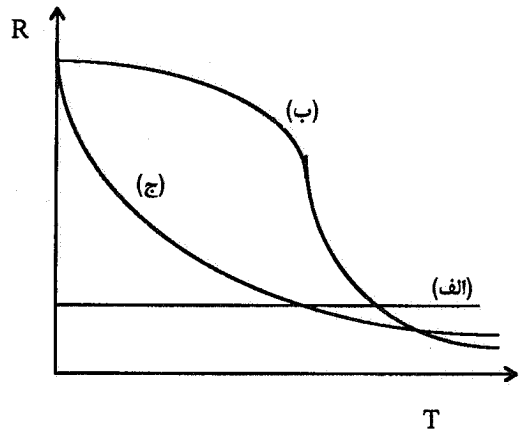
به طوری که A، B، C و T_0 پارامترهای ثابت هستند.

با مقادیر ثابت $T_0=1000$، $A=1$، $B=0.0055$، $C=5$ و $f_0=0.97$ ضریب کاهش دما) شکل توابع (الف)، (ب) و (ج) تقریباً به صورت زیر است:

جدول ۱- ارزیابی چند تابع برای R

شمارهٔ دستهٔ مسئله	$m\text{-}n_1\text{-}n_2$	t_1	t_2	t_3
۱	۷-۴-۵	۱/۰۱	۰/۸۰	۱/۱۲
۲	۱۰-۵-۱۰	۵/۹۵	۱۰/۸۹	۳/۵۱
۳	۱۵-۷-۱۶	۲۰/۲۲	۲۹/۶۱	۱۳/۹۶
۴	۲۰-۱۰-۱۵	۲۱/۹۰	۲۴/۹۴	۱۹/۷۰
۵	۲۲-۷-۱۴	۳۴/۳۲	۳۷/۶۵	۳۱/۳۸

t_1، t_2 و t_3 عبارت‌اند از متوسط زمان رسیدن به جواب مورد نظر وقتی که به ترتیب از R_1، R_2 و R_3 در الگوریتم استفاده می‌شود.



شکل ۱

برای تعیین تابع مناسب R پنج سری مسئله با اندازه‌های مختلف (در هر اندازه هفت مسئله) حل شده است و متوسط زمان رسیدن به جواب مورد نظر در حالتهای مختلف، در جدول (۱) ارائه شده است.

با توجه به نتایج ارائه شده در جدول (۱) در چهار سری از مسائل حل شده، تابع (ج) نسبت به توابع (الف) و (ب)، الگوریتم را در حل مسائل مورد نظر کاراتر کرده است، و فقط در سری اول تابع (ب) کاراتر بوده است. بنابراین تابع (ج) برای تعیین احتمال جستجوی همسایگی یک جواب نشدنی SA در هر مرحله انتخاب می‌شود.

تابع R تعیین شده دارای پارامترهای T_0، A و B است. با بررسی اجمالی این تابع مشخص شد که A تعیین کنندهٔ مقدار R در زمان صفر است و پارامترهای T_0 و B وابسته به هم بوده و تعیین می‌کنند که مقدار R پس از گذشت مدت زمانی، در چه مقداری ثابت باقی می‌ماند. بنابراین، دو پارامتر T_0 و R همزمان با هم مورد بررسی قرار گرفته است.

برای تعیین مقدار مناسب A پارامترهای دیگر ثابت فرض شده و ده مسئله با اندازهٔ ۱۲-۷-۱۷ (یعنی $m=17$، $n_1=7$ و $n_2=12$ است) حل شده است که زمانهای رسیدن به جواب مورد نظر در جدول (۲) ارائه شده است. با توجه به اطلاعات جدول (۲)، بهترین مقدار برای A عدد ۰/۸ است، چون با این مقدار متوسط زمان رسیدن به جواب کمتر از بقیهٔ حالتهاست.

برای تعیین مقدار مناسب پارامترهای T_0 و B، بررسیهای اولیه نشان داده است که مقدار مناسب حاصلضرب $B\times T_0$ باید در فاصلهٔ ۱/۵ و ۳ باشد. بنابراین برای مقادیر $B\times T_0$ برابر با ۱/۵، ۲، ۲/۵ و ۳ ده مسئله با اندازهٔ ۱۵-۶-۱۵ حل شده است که زمان رسیدن به جواب مورد نظر در جدول (۳) ارائه شده است. نتایج ارائه شده در جدول (۳) نشان می‌دهد که مقدار مناسب $B\times T_0$ برابر عدد دو است، یعنی برای $T_0=1000$ باید $B=0.002$ در نظر گرفته شود.

۴-۲- تعیین پارامترهای دیگر الگوریتم

پارامترهایی مانند ضریب تغییر دما (f_0)، تعداد تکرارها با هر دما و نحوهٔ استفاده از همسایگی با مراتب مختلف از عوامل مؤثر در کارایی الگوریتم هستند که در اینجا مورد بررسی قرار می‌گیرند.

۴-۲-۱ ضریب تغییر دما (f_0)

تابع تغییر دما در این الگوریتم در هر مرحله به صورت ضریبی از دمای مرحلهٔ قبل در نظر گرفته شده است. در اینجا تعیین مقدار

جدول ۲- نتایج اثر پارامتر A بر کارایی الگوریتم

شمارهٔ مسئله	مقدار تابع هدف سطح یک	A=1.0	A=0.9	A=0.8	A=0.7	A=0.6
۱	-۳۳/۹۵	۸/۶۸	۱۱/۲۶	۷/۸۰	۷/۹۶	۵/۴۹
۲	۶۰۵/۰۶	۶/۲۶	۳/۹۵	۴/۱۷	۱/۷۰	۱/۳۷
۳	۴۷۶/۹۴	۱/۴۸	۱/۶۵	۱/۷۰	۴/۵۶	۱۰/۳۲
۴	۲۵۹/۴۵	۲۴/۶۷	۱۳/۸۴	۱۲/۶۳	۱۲/۱۴	۱۱/۹۲
۵	۳۱۱/۸۹	۴/۸۳	۰/۵۵	۰/۵۵	۰/۵۵	۲/۵۸
۶	۲۱۸/۸۷	۱۹/۸۳	۲/۴۷	۲/۴۷	۲/۲۴	۱۳/۶۳
۷	۲۳۹/۱۶	۱/۹۲	۱۵/۹۳	۴/۳۴	۱/۹۲	۱/۹۷
۸	۶۱۶/۹۸	۸/۵۷	۵/۵۵	۵/۵۹	۱۸/۹۵	۹/۳۴
۹	۴۹۷/۹۵	۹/۹۴	۱۷/۱۴	۵/۹۸	۶/۲۶	۱۸/۳۷
۱۰	۳۱۴/۶۷	۴/۵۶	۴/۴۵	۴/۴۵	۸/۸۵	۸/۹۰
متوسط زمان حل		۹/۰۷	۷/۶۸	۴/۹۷	۶/۵۱	۸/۳۹

جدول ۳- نتایج بررسی اثر T_0 و B بر کارایی محاسباتی الگوریتم

شمارهٔ مسئله	مقدار تابع هدف سطح یک	$B\times T_0=3.0$	$B\times T_0=2.5$	$B\times T_0=2.0$	$B\times T_0=1.5$
۱	۱۷۳/۴۴	۳/۱۹	۳/۱۳	۲/۹۶	۴/۲۸
۲	۲۹۲/۲۱	۷/۲۵	۱۹/۳۴	۵/۷۱	۱۱/۱۵
۳	۴۰۱/۶۹	۱/۶۵	۱/۵۹	۱/۶۴	۴/۶۱
۴	۲۷۹/۶۷	۱/۳۷	۱/۷۷	۱/۳۱	۲۲/۸۵
۵	۲۵۶/۲۳	۱۰/۵۵	۵/۴۹	۲۶/۰۹	۱۵/۴۳
۶	۶۴۷/۶۴	۲۶/۱۵	۱۸/۵۳	۹/۲۸	۱۰/۱۰
۷	۳۵۷/۰۰	۱۱/۵۳	۱۰/۸۲	۷/۳۱	۵/۳۸
۸	۹۵/۶۷	۳/۹۵	۶/۷۵	۴/۳۰	۴/۳۹
۹	۴۴۶/۳۹	۲۲/۰۳	۲۷/۱۹	۱۱/۳۷	۱۱/۳۲
۱۰	۹۷۱/۹۷	۱۵/۱۰	۱۵/۲۱	۱۸/۶۲	۱۹/۷۲
متوسط زمان حل		۱۰/۲۸	۱۰/۹۸	۸/۸۶	۱۰/۹۲

مناسب ضریب تغییر دما (f_0) مورد نظر است. بدین منظور ده مسئله با اندازهٔ ۱۲-۷-۱۷، به ازای مقادیر مختلف f_0 حل شده است که زمان رسیدن به جواب مورد نظر در حالتهای مختلف در جدول (۴) ارائه شده است.

با توجه به نتایج ارائه شده در جدول (۴) حالتهایی که f_0 برابر ۰/۹۶ و ۰/۹۷ بوده متوسط زمان کمتری به دست آمده است. یعنی

جدول ۴- اثر ضریب تغییر دما بر کارایی الگوریتم

شمارهٔ مسئله	مقدار تابع هدف سطح یک	f_0=0.94	f_0=0.95	f_0=0.96	f_0=0.97	f_0=0.98	f_0=0.99
۱	-۳۳/۹۵	۷/۸۶	۱۰/۳۳	۷/۵۸	۷/۸۰	۸/۸۴	۴/۹۴
۲	۶۰۵/۰۶	۱/۶۵	۱/۵۹	۴/۲۸	۴/۱۷	۶/۲۶	۶/۵
۳	۴۷۶/۹۴	۷/۹۶	۷/۳۶	۱/۷۰	۱/۷۰	۱/۶۵	۱/۶۵
۴	۲۵۹/۴۵	۱۲/۴۲	۱۲/۵۳	۱۲/۲۵	۱۲/۶۳	۱۷/۸۱	۲۵/۸۷
۵	۳۱۱/۸۹	۰/۵۵	۰/۶۰	۰/۶۰	۰/۵۵	۰/۵۵	۰/۶۰
۶	۲۱۸/۸۷	۱۳/۹	۱۱/۸۱	۲/۴۷	۲/۴۷	۲/۶۹	۲۱/۷۵
۷	۲۳۹/۱۶	۱/۱۵	۸/۸۵	۴/۲۹	۴/۳۴	۴/۲۹	۱۲/۳۰
۸	۶۱۶/۹۸	۱۴/۱۲	۶/۴۸	۵/۹۳	۵/۵۹	۸/۸۴	۸/۸۴
۹	۴۹۷/۹۵	۲۴/۵۲	۲۱/۱۰	۶/۲۶	۵/۹۸	۵/۶۶	۳۹/۸۳
۱۰	۳۱۴/۶۷	۶/۴۸	۴/۳۹	۴/۳۹	۴/۴۵	۴/۳۹	۱۴/۰۶
متوسط زمان حل		۹/۰۶	۸/۵۰	۴/۹۷	۴/۹۷	۶/۱۰	۱۳/۶۳

به ازای این مقادیر برای f_0 کارایی محاسباتی الگوریتم بالاتر خواهد بود.

۴-۲-۲ تعداد تکرارها در هر دما (n_0)

پارامتر دیگری که ممکن است کارایی محاسباتی الگوریتم را تحت تأثیر قرار دهد، تعداد تکرارها در هر دماست. برای بررسی اثر این عامل و تعیین مقدار مناسب برای آن چهار گروه مسئله که در هر گروه اندازهٔ مسئله ثابت است، حل شده است. در اینجا کلیه پارامترها ثابت فرض شده و فقط تعداد تکرارها تغییر یافته است.

متوسط زمان رسیدن به جواب مورد نظر برای تعداد تکرارهای مختلف در جدول (۵) ارائه شده است.

اطلاعات این جدول نشان می‌دهد که در سه گروه از مسائل حل شده، تعداد تکرار چهار، کارایی بیشتری داشته است. فقط در گروه سوم کارایی تعداد تکرار سه بیشتر از تعداد تکرار چهار است، که این اختلاف نیز بسیار جزیی است. به همین دلیل با توجه به مسائل حل شده بهترین مقدار برای n_0، چهار است.

۴-۲-۳ نحوهٔ استفاده از همسایگیها با مراتب مختلف

در این الگوریتم همسایگی با مراتب مختلف تعریف شده است و می‌توان برای انتخاب یک جواب در همسایگی جواب فعلی از همسایگی با مراتب مختلف استفاده کرد. برای تعیین نحوهٔ استفاده از همسایگیها با مراتب مختلف بررسیهای اولیه نشان داد، استفاده از یک همسایگی مشخص به طور ثابت در طول اجرای الگوریتم، مخصوصاً برای همسایگیهای با مرتبهٔ سه و بالاتر برخی از مسائل را در زمان قابل قبول به جواب مورد نظر نمی‌رساند. به همین دلیل برای بررسی دقیقتر، حالتهای زیر در نظر گرفته شده است:

الف - همسایگی مرتبهٔ یک

ب - همسایگی مرتبهٔ دو

ج - همسایگیهای مرتبهٔ یک و دو، تواماً و با احتمال به کارگیری یکسان

د - همسایگیهای مرتبهٔ یک، دو و سه، تواماً و با احتمال به کارگیری یکسان.

در جدول (۶) زمانهای به دست آمده برای حل ده مسئله با اندازهٔ ۹-۴-۱۰ برای حالتهای فوق‌الذکر ارائه شده است. همان گونه که از اطلاعات این جدول پیداست متوسط زمان رسیدن به جواب مورد نظر برای حالت (ج)، ۲/۳۶ ثانیه است، که از بقیه حالتها کمتر است.

بنابراین با توجه به مسائل حل شده بهترین نحوهٔ استفاده از

جدول ۵- اثر تعداد تکرارها در هر دما بر کارایی الگوریتم

شمارهٔ دستهٔ مسئله	$m\text{-}n_1\text{-}n_2$	$n_0=3$	$n_0=4$	$n_0=5$	$n_0=6$	$n_0=7$
۱	۷-۴-۵	۱/۲۶	۱/۲۳	۱/۴۵	۱/۷۳	۱/۷۴
۲	۱۰-۴-۹	۷/۳۵	۳/۰۸	۴/۴۶	۴/۰۴	۶/۵۹
۳	۱۵-۷-۱۰	۵/۰۴	۵/۶۵	۶/۶۲	۶/۷۶	۶/۹۴
۴	۲۰-۱۰-۱۲	۱۲/۵۳	۱۱/۳۵	۱۲/۳۲	۱۴/۵۸	۱۲/۴۶

جدول ۶- نتایج حاصل از بررسی اثر نحوهٔ استفاده از همسایگیها با مراتب مختلف

شمارهٔ مسئله	مقدار تابع هدف سطح یک	حالت (الف)	حالت (ب)	حالت (ج)	حالت (د)
۱	-۳۸۲/۴۲	۸/۱۳	۴/۸۹	۱/۸۱	۳/۹۵
۲	۲۷/۸۶	۰/۳۳	۲/۷۵	۱/۱۵	۲/۰۹
۳	-۴۶/۱۹	۳/۸۵	۲/۴۷	۳/۶۳	۲/۵۳
۴	۲۳۵/۸۵	۶/۴۳	۱/۴۳	۶/۷۵	۴/۴۵
۵	۶۱۶/۸۱	۱۰/۷۶	۸/۵۷	۲/۱۴	۰/۲۲
۶	۱۳۴/۵۱	۵/۶۰	۱۰/۰	۳/۸۵	۸/۸۴
۷	۶۳۴/۲۳	۲/۸۰	۰/۵۵	۳/۰۲	۱/۹۸
۸	۴۱۰/۷۲	۰/۶۶	۲/۰۳	۰/۲۷	۱/۲۱
۹	۱۱۲/۵۷	۳/۱۹	۴/۳۹	۰/۹۳	۱/۶۵
۱۰	۴۶۶/۹۵	۱۲/۸۵	۰/۱۱	۰/۱۱	۱۳/۳۵
متوسط زمان حل		۵/۴۶	۳/۷۲	۲/۳۶	۴/۰۳

همسایگی با مراتب مختلف، استفاده از مرتبه‌های یک و دو با احتمالهای به کارگیری یکسان است.

۴-۳- مقایسهٔ روش GA ارائه شده توسط ماتیو و همکارانش با روش SA پیشنهادی

ماتیو و همکارانش [۱] روشی بر پایه الگوریتم ژنی برای حل BLP ارائه کرده‌اند. در این روش هر کروموزوم یک رشته اعداد حقیقی به طول n_1+n_2 است. n_1 عدد نخستین این رشته، مقادیر متغیرهای تحت کنترل تصمیم گیرندهٔ رهبر و n_2 عدد باقیماندهٔ مقادیر متغیرهای تحت کنترل تصمیم گیرندهٔ پیرو هستند. در واقع هر کروموزوم شدنی در این روش یک جواب قابل دستیابی BLP است اما این جواب لزوماً یک نقطه رأسی فضای قابل دستیابی نیست.

در الگوریتم GA ارائه شده دو روش برای تولید جمعیت نسل جدید استفاده شده است. روش اول تولید تصادفی مقدار متغیرهای تصمیم گیرندهٔ رهبر و به دست آوردن متغیرهای تصمیم گیرندهٔ پیرو با استفاده از حل مسئلهٔ سطح دوم است و روش دیگر استفاده از فرایند جهش است که در این روش یک یا چند جزء از n_1 جزء اول یک کروموزوم موجود را تغییر داده و n_2 جزء آخر آن، با حل مسئلهٔ سطح دوم پس از اعمال تغییرات به دست می‌آید. در فرایند جهش این الگوریتم حدود تغییر اجزا در هر مرحله تغییر می‌کند و به تدریج کاهش پیدا می‌کند، وقتی این حدود تغییر از مقدار کوچکی مانند ε

جدول ۷- مقایسهٔ روش پیشنهادی با روش ماتیو و همکارانش

شمارهٔ دستهٔ مسئله	$m\text{-}n_1\text{-}n_2$	t_{SA}	Δ_{SA}	t_{GA}	Δ_{GA}
۱	۶-۱-۳	۰/۶۵	۰/۰۰	۱/۷۶	۱۸/۰۷
۲	۵-۴-۵	۰/۷۸	۰/۰۰	۲۴/۵۵	۳۸/۷۷
۳	۷-۴-۵	۱/۱۴	۰/۰۰	۲۳/۹۵	۴۹/۷۶
۴	۸-۵-۶	۲/۱۰	۰/۰۰	۳۶/۰۲	۴۸/۹۶
۵	۹-۶-۸	۵/۳۶	۰/۰۰	۷۰/۸۲	۵۲/۷۱
۶	۱۰-۸-۵	۴/۲۱	۰/۰۰	۸۷/۹۲	۷۹/۱۴
۷	۱۱-۷-۱۰	۷/۳۲	۲/۵۰	۱۹۲/۵۶	۵۰/۸۸
۸	۱۵-۱۰-۱۰	۱۸/۳۳	۱/۳۱	-	-
۹	۲۵-۱۵-۱۵	۳۴/۳۴	۰/۲۵	-	-
۱۰	۳۰-۱۵-۲۰	۴۳/۷۴	۱/۱۰	-	-

کمتر شود الگوریتم متوقف شده و بهترین جواب به دست آمده را به عنوان جواب BLP گزارش می‌کند.

در این مقاله کارایی با دو عامل زمان انجام محاسباتی (واحد ثانیه) و کیفیت جواب به دست آمده که به صورت زیر سنجیده می‌شود مورد ارزیابی قرار می‌گیرد.

$$\Delta = (F^*\text{-}F)/F^*)\times 100$$

هر چقدر Δ برای یک جواب کمتر باشد آن جواب بهتر است.

برای مقایسهٔ دو الگوریتم ده گروه از مسائل با اندازه‌های مختلف که در هر گروه پنج مسئله قرار دارد حل شده است و نتایج به دست آمده در جدول (۷) ارائه شده است. با توجه به داده‌های جدول (۷)، در شش گروه اول از مسائل، مقدار Δ_{SA} برای روش SA پیشنهادی صفر است، یعنی روش پیشنهادی جواب بهینه را به دست آورده است. در چهار گروه آخر نیز مقدار Δ_{SA} کوچکتر یا مساوی ۲/۵ درصد است. در حالی که برای روش GA در هفت گروه اول از مسائل مقدار Δ_{GA} حداقل ۱۸/۰۷ درصد است و در سه گروه آخر، الگوریتم GA به ترتیب در زمانهای ۲۰۰، ۳۰۰ و ۵۰۰ ثانیه هیچ جوابی به دست نیاورده است.

همچنین از نظر زمان رسیدن به جواب نیز روش SA پیشنهادی در زمانهای بسیار کمتری به جواب رسیده است.

بنابراین روش SA با توجه به مسائل حل شده هم از نظر زمان رسیدن به جواب و هم از نظر کیفیت جواب به دست آمده برتری مشخصی نسبت به روش GA ارائه شده توسط ماتیو و همکارانش دارد.

۵- خلاصه و نتیجه‌گیری

روشهای مدرن ابتکاری برای کاهش پیچیدگی محاسباتی ارائه شده‌اند و می‌توانند در حل BLP که یک مسئلهٔ Np-hard است مؤثر باشند.

هدف از تهیه این مقاله توسعهٔ روشی براساس الگوریتم SA برای حل BLP بوده است. در روش توسعه داده شده، ابتدا شرایط کوهن - تاکر برای مسئلهٔ سطح دوم نوشته می‌شود تا مسئله به صورت یک برنامه ریزی یک هدفی با محدودیتهای مکمل تبدیل شود. سپس با تعریف یک جواب SA، برای مدل تبدیل شده، و بیان چند قضیه اساسی، روش SA برای حل BLP توسعه داده شده است. این روش نقاط رأسی فضای قابل دستیابی را جستجو و ارزیابی می‌کند.

پس از تخمین مقدار مناسب پارامترهای الگوریتم توسعه داده شده با حل مسائل متعدد، این روش با روش GA ارائه شده توسط ماتیو و همکارانش [۱] مقایسه شده است. نتایج محاسبات نشان می‌دهند که روش SA پیشنهادی هم از نظر زمان محاسبات و هم از نظر کیفیت جواب به دست آمده برتری زیادی نسبت به روش ماتیو و همکارانش دارد.

در زمینهٔ استفاده از روشهای مدرن ابتکاری برای حل BLP تحقیقاتی انجام شده است. علاوه بر مقالهٔ حاضر، مقاله‌ای که در آن از الگوریتم ژنی برای حل BLP استفاده شده است، توسط مؤلفان این مقاله آماده شده است که در دست چاپ است. همچنین تحقیقات آتی برای به کارگیری روشهای مدرن ابتکاری دیگر و تعیین بهترین روش مدرن ابتکاری برای حل BLP ادامه دارد.

واژه نامه

1. bilevel programming
2. Simulated Annealing (SA)
3. computational effeciency
4. modern heuristic method
5. slack
6. surplus
7. dual
8. accessible region
9. vertex points
10. vertex enumeration
11. fuzzy approach
12. branch and bound
13. kth-best
14. local optima
15. Grid Search Algorithm (GSA)
16. Bicriteria Programming (BCP)
17. complementary constraints
18. parametric complementary pivot
19. Linear Complementary Problem (LCP)
20. penalty function
21. noncooperative principle
22. Genetic Algorithm (GA)
23. chromosome

قدردانی

ضمن تشکر فراوان از داوران محترم این مقاله که نظرات ارزندهٔ آنها باعث بهبود کیفیت مقاله شد، لازم می‌دانیم از آقایان دکتر حسام‌الدین ذگردی و دکتر محمدمهدی سپهری به خاطر راهنمایی آنها در انجام این تحقیق سپاسگزاری کنیم.

مراجع

1. Mathiue, R., Pittard, L., and Anandalingam, G., "Genetic Algorithm Based Approach to Bilevel Linear Programming," *Recherche Operationnelle / Operations Research*, Vol. 28, No. 1, pp. 1-21, 1994.
2. Falk, J. E., "A Linear Max-Min Problem," *Math. Prog.,* Vol. 5, pp. 169-188, 1973.
3. Bialas, W. F., and Karwan, M. H., "On Two-level Optimization," *IEEE Trans. Autom. Control,* Vol. 27, No. 1, pp. 11-214, 1982.
4. Bialas, W. F., and Karwan, M. H., "Two-level Linear Programming," *Mgmt Sci.,* Vol. 30, pp. 1004-1020, 1983.
5. Bard, J. F., "An Efficient Point Algorithm for a Linear Two-Stage Optimization Problem," *Ops. Res.,* Vol. 38, pp. 670-684, 1983.
6. Ünlü, G., "A Linear Bilevel programming Algorithm Based on Bicritria Programming," *Comput. Ops. Res.,* Vol. 14, pp. 173-179, 1987.
7. Candler, W., "A Linear Bilevel Programming Algorithm: A Coment," *Comput. Ops. Res.,* Vol. 15, No. 3, pp. 297-298, 1983.
8. Clark, P. A., and Westerberg, A. W., "A Note on the Optimality Conditions for the Bilevel Programming Problem," *Nav. Res. Logist. Q.,* Vol. 35, pp. 413-418, 1988.
9. Wen, U. P., and Hsu, S. T., "A Note on a Linear Bilevel Programming Algorithm Based on Bicriteria Programming," *Comput Ops. Res.,* Vol. 16, No. 1, pp. 79-83, 1989.
10. Ben-Ayed, O., and Blair, C. E., "Computational Difficulties of Bilevel Linear Programming," *Ops. Res.,* Vol. 38, pp. 556-560, 1990.
11. Marcotte, P., and Savard, G., "A Not on the Pareto Optimality of Solutions to the Linear Bilevel Programming Problem," *Comput. Ops. Res.,* Vol. 18, No. 4, pp. 355-359, 1991.
12. Fortury-Amat, J., and McCarl, B., "A Representation and Economic Interpretation of a Two-level programming Problem," *J. Opl. Res. Soc.,* Vol. 32, pp. 783-792, 1981.
13. Bard, J. F., and Moore, J. T., "A Branch and Bound Algorithem for the Bilevel Programming Problem," *SIAM J. Sci. Stat. Comput.,* Vol. 11, No. 2, pp. 281-292, 1990.
14. White, D. J., and Anandalingam, G., "A Penalty

Function Approach for Solving Belevel Linear Programs," *Journal of Global Optimization,* Vol. 3, pp. 397-419, 1973.

15. Anandalingam, G., and White, D. J., "A Solution for the Linear Static Stackelberg Problem Using Penalty Functions," *IEEE Trans. Autom. Control,* Vol. 35, pp. 1170-1173, 1990.
16. Shih, S. H., Lai, Y. J., and Lee, E. S., "Fuzzy Approach for Multilevel Programming Problem," *Comput. Ops. Res.,* Vol. 23, No. 1, pp. 773-791, 1983.
17. Sakava, M., Nishizaki, I., and Uemura, Y., "Interactive Fuzzy programming for Multilevel Linear Programming problem," *Comput Math. Applic.,* Vol. 6, pp. 71-86, 1997.
18. Sahin, K. H., and Cirit, A. R., "A Dual Temperature Simulated Annealing Approach for Solving Bilevel Programming Problems," *Computers and Chemical Engineering,* Vol. 23, pp. 11-25, 1998.
19. Hejazi, S. R., Memariani, A., Jahanshahloo, G. R., and Sepehri, M. M., "Bilevel Programming Solution by Genetic Algorithm," 1999, (Communicated).

[DOR: 20.1001.1.22287698.1379.19.2.6.6]