حل برنامه ریزی دو سطحی[۱] با الگوریتم جستجوی ممنوع[۲]

سیدرضا حجازی

دانشجوی دکترای دانشکده مهندسی صنایع و سیستم - دانشگاه صنعتی اصفهان

عزیز اله معماریانی

استادیار بخش مهندسی صنایع - دانشکدهٔ فنی و مهندسی - دانشگاه تربیت مدرس

محمد مهدی سپهری

استادیار بخش مهندسی صنایع - دانشکدهٔ فنی و مهندسی - دانشگاه تربیت مدرس

غلامرضا جهانشاهلو

استاد بخش ریاضی - دانشگاه تربیت معلم

(تاریخ دریافت ۷۹/۶/۶، تاریخ تصویب ۸۰/۵/۳۰)

چکیده

برنامه ریزی دو سطحی ابزاری برای مدلسازی مسئله تصمیم گیری غیر متمرکز است. که در آن تصمیم گیرندهٔ سطح یک و دو بترتیب رهبر و پیرو گفته می شوند. ثابت شده است که مسئله برنامه ریزی دو سطحی یک مسئلهٔ Np_hard است. روشهای زیادی برای حل این مسئله ارایه شده است؛ اما کارآیی محاسباتی[۳] آنها در حدی نیست که بتوانند مسائل بزرگ را حل کنند. در این مقاله سعی شده است روشی بر اساس جستجوی ممنوع برای حل مسئله برنامه ریزی دو سطحی توسعه داده شود. این روش از روشهای فراابتکاری[۴] است که می تواند 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 ارایه شده توسط ماتیو و همکارانش[۱] مقایسه شده است.

واژه های کلیدی : برنامه ریزی ریاضی، برنامه ریزی دوسطحی، روشهای فراابتکاری، جستجوی ممنوع

مقدمه

بیشتر مدلهای ریاضی شامل یک تصمیم گیرنده و یک تابع هدف هستند، که برای برنامه ریزی متمرکز بکار می روند؛ اما برنامه ریزی ریاضی دو سطحی برای تصمیم گیری غیرمتمرکز توسعه داده شده است. در برنامه ریزی دو سطحی(BLP) که تصمیم گیرندهٔ سطح یک آن را رهبر و سطح دو آن را پیرو می گوییم، هر تصمیم گیرنده سعی می کند تابع هدف خود را بدون توجه به هدف قسمت دیگر بهینه کند؛ اما تصمیم هر تصمیم گیرنده بر مقدار تابع هدف و فضای تصمیم گیری سطح دیگر اثر می گذارد. شکل ریاضی برنامه ریزی دوسطحی بصورت زیر است:

$$\begin{aligned} &\underset{x}{\text{Max}} \quad F(x,y) = c_1x + d_1y \\ &\underset{y}{\text{Max}} \quad f(x,y) = c_2x + d_2y \\ &\text{s.t}: \begin{cases} A_1x + A_2y \le b \\ x, y \ge 0 \end{cases} \end{aligned} \qquad (1)$$

که در آن F(x,y) تابع هدف رهبر و f(x,y) تابع هدف پیرو می باشد، همچنین x و y متغیرهای را نشان می دهند که بترتیب تحت کنترل تصمیم گیرندهٔ سطح یک و دو هستند.

نقطهٔ (x_0, y_0) را یک نقطهٔ شدنی می گوییم اگر در محدودیتهای مدل (۱) صدق کند و آن را یک نقطهٔ قابل دستیابی برای تصمیم گیرنده سطح یک می گوییم اگر

برای $x = x_0$، y_0 جواب بهینهٔ مسئلهٔ سطح دوم باشد. مسئله سطح دوم بصورت زیر تعریف می شود[۳] .

$$\max \; d_2 y + c_2 x_0$$
$$\begin{cases} A_2 y \le b - A_1 x_0 \\ y \ge 0 \end{cases} \qquad (۲)$$

که در آن x_0 توسط سطح یک تعیین شده است. همچنین مجموعهٔ تمام نقاط قابل دستیابی را ناحیهٔ قابل دستیابی[۵] مسئلهٔ BLP می گوییم که زیر مجموعه ای از مجموعه جوابهای شدنی BLP است و لزوما" یک مجموعهٔ محدب نیست . هدف از حل مسئلهٔ BLP، بدست آوردن نقطه ای از فضای قابل دستیابی است که به ازاء آن مقدار تابع هدف سطح یک، روی ناحیهٔ قابل دستیابی بهینه شده باشد. از ویژگیهای مهم ناحیهٔ قابل دستیابی این است که، مجموعهٔ نقاط رأسی[۶] آن، زیر مجموعهٔ نقاط رأسی فضای شدنی است و جواب بهینهٔ BLP نیز یکی از این نقاط رأسی است.

روشهای ارایه شده برای حل این مسئله را می توان بصورت زیر دسته بندی کرد :

الف - روشهایی بر اساس شمارش رئوس[۷]

ب - روشهایی بر اساس شرایط کوهن-تاکر

پ - نگرش فازی[۸] برای حل BLP

ت - روشهای فراابتکاری برای حل BLP

الف - روشهایی بر اساس شمارش رئوس

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

اوّلین روش بر این اساس را فالک [۲] برای مسئلهٔ Max_Min که یک مورد خاص از BLP است، توسعه داده است. در این روش نقاط رأسی مسئله سطح یک، با یک روش شاخه وکران[۹] جستجو می شود. بیالاس و کاروان[۳] الگوریتمی تحت عنوان Kامین بهترین[۱۰] ارایه کرده اند که یکی از روشهای مهم توسعه داده شده برای حل این مسئله است؛ در این روش فرآیند جستجو از ناحیهٔ غیر قابل دستیابی شروع شده و اوّلین نقطهٔ قابل دستیابی که پیدا شود، الگوریتم متوقف می شود و این نقطه جواب بهینهٔ مسئلهٔ BLP است.

بیالاس و کاروان [۴] همچنین روشی را پیشنهاد کرده اند که از یک نقطهٔ رأسی ناحیهٔ قابل دستیابی جستجو را آغاز می کند؛ اما این روش ممکن است در یک جواب بهینهٔ نسبی[۱۱] متوقف شود.

الگوریتم دیگری بنام GSA [۱۲] توسط برد [۵] و بنام BCP [۱۳] توسط انلو [۶] بر پایهٔ برنامه ریزی خطی دو معیاری 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 دو سطحی کاملا" متفاوت بوده و نمی توان آنها را بسادگی بهم ارتباط داد.

ب - روشهایی بر اساس 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 می شود. آمات و مک کارل [۱۲] برای حل این مسئله محدودیتهای مکمل[۱۴] بوجود آمده را به محدودیتهای خطی با متغیرهای صفر ویک تبدیل کرده و مسئله را حل می کنند. بیالاس و کاروان [۴،۳] الگوریتم چرخش لولایی مکمل پارامتری[۱۵] (PCP) را توسعه داده اند، در این روش تابع هدف سطح یک بصورت محدودیتی بشکل، $c_1 x + d_1 y \le \alpha$ به محدودیتهای مسئله اضافه شده و به یک مسئلهٔ مکمل خطی[۱۶] (LCP تبدیل می شود؛ اگر این مسئله شدنی باشد، با افزایش α می توان به یک جواب بهتر دست یافت؛ اگر در مرحله ای، LCP شدنی نباشد، جواب شدنی مرحلهٔ قبل جواب مسئلهٔ مورد نظر با این روش است.

برد و مور[۱۳] نیز روش شاخه و کرانی برای حل این مسئله توسعه داده اند. همچنین وایت و آناندالینگام [۱۴] و آناندالینگام و وایت [۱۵] با اضافه کردن یک تابع

جریمه[۱۷] بعنوان بخشی از تابع هدفسطح یک، جهت ارضاء محدودیتهای مکمل، مسئلهٔ مورد نظر را حل کرده اند.

پ - نگرش فازی برای حل BLP

با این نگرش در سال ۱۹۹۶ روشی توسط شیه و همکارانش [۱۶] ارایه شده است که پس از آن ساکاوا، نیشی زاکی و یومورا[۱۷] این روش را با اصلاحاتی ارایه کرده اند. در این روش اصل عدم همکاری[۱۸] که یکی از اصول اساسی BLP است حذف شده است و در واقع روشی برای رسیدن به یک جواب توافقی ارایه شده است.

ت - روشهای فرا ابتکاری برای حل BLP

در این زمینه در سال ۱۹۹۴ ماتیو، پیتارد و آناندالینگام [۱] روشی بر اساس الگوریتم ژنی[۱۹] ارایه کرده اند، روش ارایه شده، تمام ناحیهٔ قابل دستیابی را جستجو می کند و هر کروموزوم[۲۰] شدنی آن معرف یک جواب قابل دستیابی BLP است.

ساهین و سیریت[۱۸] روشی بر اساس سیمیولیتد اننیلینگ ارایه کرده اند. ایدهٔ اصلی این روش بسط فضای جستجو با فازی سازی مسئله وجستجوی تصادفی بکمک زنجیرههای مارکوف[۲۱] است. آنها مسائل کم و کوچکی را حل و گزارش کرده اند که گزارش ارایه شده نشان می دهد، این روش کارآیی محاسباتی مناسبی برای حل برنامه ریزی دوسطحی خطی را ندارد. بعنوان نمونه مسئله 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 می کند؛ اما بهرحال این روش قادر است هر برنامه ریزی دو سطحی مانند خطی، غیر خطی یا عدد صحیح را حل کند.

روشی نیز بر اساس الگوریتم ژنی توسط حجازی وهمکارانش[۱۹] توسعه داده شده است که در این روش هر کروموزوم شدنی معرف یک نقطهٔرأسی ناحیهٔ قابل دستیابی است و نسبت به روش ماتیو، پیتارد و آناندالینگام [۱] فضای بسیار محدود تری را جستجو می کند .

روشی که در این مقاله ارایه می شود بر اساس TS توسعه داده شده است و نقاط رأسی فضای قابل دستیابی را جستجو می کند.

بکارگیری TS برای حل BLP

معرفی روش جستجوی ممنوع

این روش بطور کلی شامل دوقسمت است که عبارتند از :

الف - فرآیند رفتن به یک جواب بهینهٔ نسبی

ب - فرآیند خروج از نقطهٔ بهینهٔ نسبی

در روش جستجوی ممنوع ابتدا با استفاده از شیوه های موجود، یک جواب شدنی اولیه برای مسئله بدست می آید. سپس با اعمال حرکتهای تعریف شده، مجموعهٔ جوابهای همسایگی جواب فعلی ارزیابی شده و حرکتی که بیشترین بهبود در تابع هدف می دهد، اعمال می شود تا جواب شدنی بعدی بدست آید و بدلیل اینکه جوابهای تکراری تولید نشود و الگوریتم در یک حلقهٔ بسته قرار نگیرد، این حرکت در لیست حرکتهای ممنوع قرار می گیرد. با تکرار این فرآیند پس از چند مرحله یک جواب بهینهٔ نسبی بدست می آید. این فرآیند، فرآیند رفتن به یک جواب بهینهٔ نسبی گفته می شود.

بمنظور اینکه الگوریتم در یک نقطهٔ بهینهٔ نسبی متوقف نشود، هر حرکت ممنوع، پس از چند مرحله اجرای الگوریتم دو باره مجاز می شود؛ ویا اینکه پس از چند مرحله اجرای الگوریتم، بطریقی که به ماهیت مسئله بستگی دارد فضای جستجو تغییر داده می شود تا الگوریتم از نقطهٔ بهینهٔ نسبی خارج شود و به نقطهٔ بهینهٔ نسبی دیگر برود، این فرآیند، فرآیند خروج از نقطهٔ بهینهٔ نسبی گفته می شود. با تکرار دو فرایند فوق نهایتا" الگوریتم به سمت جواب بهینهٔ مطلق خواهد رفت.

تعاریف و قضایای اولیه لازم برای حل BLP

مدل برنامه ریزی دو سطحی (۱) را در نظر می گیریم؛ در ابتدا شرایط کوهن-تاکر را برای مسئلهٔ سطح دوم نوشته و در مدل جایگزین می کنیم. در این صورت خواهیم داشت[۳،۴]:

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

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

(۳)

مدل (۳) یک برنامه ریزی ریاضی غیر خطی است. اگر $y\,,\ v\,,\ z\,,\ u$ ها را بگونه ای محدود کنیم که محدودیتهای $uz = 0$ و $vy = 0$ برقرار باشد، می توانیم 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.

$$\text{Max} \quad c_1x + d_1y^*$$

$$\text{s.t}: \begin{cases} A_1x + A_2y^* + u^* = b \\ z^*A_2^* - v^* = d_2 \\ x, y^*, z^*, u^*, v^* \geq 0 \end{cases}$$

(۴)

برای ساختن مدل (۴) با ویژگی ذکر شده (هر جواب شدنی مدل (۴) یک جواب شدنی مدل (۳) باشد.)، از یک رشتهٔ اعداد به طول $m + n_2$ که هر عدد (جزء) آن می تواند صفر یا یک باشد، بعنوان راهنمای تغییرات استفاده می کنیم. 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 می گیریم. اگر جزء i ام از m جزء اول عدد یک باشد، در مدل (۴) u_i را مساوی صفر قرار می دهیم. و اگر جزء i ام از m جزء اول عدد صفر باشد، در مدل (۴) z_i را مساوی صفر قرار می دهیم تا همواره u_iz_i در مدل (۴) مساوی صفر باشد. همچنین اگر جزء iام از بین n_2 جزء آخر رشتهٔ اعداد مورد نظر عدد یک باشد، در مدل (۴) y_j را مساوی صفر قرار می دهیم و اگر این جزء عدد صفر باشد، در مدل (۴) v_j را مساوی صفر قرار می دهیم در این صورت در مدل (۴)، v_jy_j همواره مساوی صفرخواهد بود. بدین ترتیب مدل (۴) بگونه ای ساخته می شود که مجموعه جوابهای شدنی آن زیر مجموعه ای از مجموعه جوابهای شدنی مدل (۳) باشد.

تعریف ۱: یک رشته اعداد بطول $m + n_2$ که هر جزء آن عدد صفر یا یک باشد را یک جواب الگوریتم TS می نامیم.

تعریف ۲: اگر مدل (۴)، متناظر یک جواب TS شدنی باشد، این جواب TS را شدنی و در غیر این صورت آن را نشدنی می گوییم.

با توجه به تعاریف فوق، فضای جوابهای TS برای مدل (۳) گسسته وتعداد جوابهای آن 2^{m+n_2} است، که برخی از آنها شدنی وبرخی نشدنی هستند.

قضیه ۱: فرض کنیم مجموعه جوابهای مدل (۳)، S باشد و N تعداد جوابهای شدنی TS باشد که فضای جوابهای شدنی مدل (۴)، مربوط به هر کدام را با $A_N, \ldots, A_2, A_1$ نشان دهیم، آنگاه داریم؛

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

اثبات: بدیهی است که هر جواب متعلق به A_i $(i = 1,2,\ldots,N)$؛ ، چون تمام محدودیتهای مدل (۳) را ارضاء می کند متعلق به S نیز است، یعنی داریم:

$$\bigcup_{i=1}^{N} A_i \subseteq S \qquad (a_1)$$

از طرف دیگر اگر جوابی مانند $(x_0, y_0, u_0, v_0, z_0)$ متعلق به S باشد؛ آنگاه به ازاء $(i = 1,2,\ldots,m)$ ؛ $u_{0i}z_{0i} = 0$ ، باید حتماً" u_i و یا z_i مساوی صفر باشد. در این صورت جواب TS مانند P را می سازیم؛ بطوریکه 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 صفرباشد و اگر u_{0i} و z_{0i} هر دو مساوی صفر باشند؛ آنگاه این جزء می تواند بطور اختیاری عدد صفر یا یک انتخاب شود.

همچنین بازاء $v_{0j}y_{0j} = 0$ $(j = 1,2,\ldots,n_2)$؛ ، حتماً" باید v_{0j} و یا y_{0j} مساوی صفر باشد. اگر y_{0j}

مساوی صفر است، در P، جزء j ام از n_2 جزء آخر را برابر عدد یک قرار می دهیم و اگر y_{0j} بزرگتر از صفراست، این جزء را صفر قرار می دهیم ودر صورتیکه v_{0j} و y_{0j} هر دو صفر باشند، جزء مورد نظر را بطور اختیاری عدد صفر یا یک انتخاب می کنیم.

بدین ترتیب P یک جواب شدنی TS خواهد بود که $(x_0, y_0, u_0, v_0, z_0)$، یک جواب شدنی مدل (۴) متناظر آن است. پس می توان نتیجه گرفت:

$$S \subseteq \bigcup_{i=1}^{N} A_i \qquad (a_2)$$

از روابط (a_1) و (a_2) نتیجه می شود که:

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

روش حل مدل (۴)

برای حل مدل (۴) متناظر یک جواب TS، می توان ابتدا این مدل را به دو مسئله کاملا" از هم جدا بصورت زیر تجزیه کرد:

$$\text{Max} \quad c_1x + d_1y^*$$

$$\text{s.t}: \begin{cases} A_1x + A_2y^* + u^* = b \\ x, y^*, u^* \geq 0 \end{cases}$$

(۵)

$$\begin{cases} z^*A_2^T - v^* = d_2 \\ z^*, v^* \geq 0 \end{cases}$$

(۶)

چون دو مسئلهٔ (۵) و (۶) هیچ متغیر مشترکی ندارند تجزیهٔ فوق مجاز است. حال اگر هر دو مسئله فوق شدنی باشد، مدل (۴) شدنی است در غیر این صورت مدل (۴) نشدنی است.

بررسی شدنی بودن یک جواب TS

برای اینکه تعیین کنیم یک جواب TS مانند P شدنی است یا نه، ابتدا مسئلهٔ (۵) متناظر آن را می سازیم و حل می کنیم، اگر نشدنی باشد آنگاه P نیز نشدنی خواهد بود. در غیر این صورت مسئله (۶)، متناظرP را حل می کنیم، اگر این مسئله شدنی باشد، آنگاه P نیز یک جواب شدنی TS خواهد بود و در غیر این صورت P یک جواب نشدنی TS است.

قضیه ۲: جواب بهینهٔ بدست آمده از مدل (۴) مربوط به یک جواب شدنی TS، متناظر یک نقطهٔ رأسی ناحیهٔ قابل دستیابی مدل (۱) است.

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

با توجه به قضیهٔ ۲، الگوریتمی که توسعه داده می شود نقاط رأسی ناحیهٔ قابل دستیابی را جستجو و ارزیابی می کند.

قضیه ۳ : اگر مسئله (۶)، برای یک جواب TS مانند P نشدنی باشد؛ هر جواب TS دیگری مانند Q که اجزاء صفر P در آن نیز صفر باشد، نشدنی است.

اثبات: فرض کنیم مسئلهٔ (۶)، متناظر P بصورت زیر باشد:

$$\begin{cases} z^*A_2^T - v^* = d_2 \\ z^*, v^* \geq 0 \end{cases}$$

(۷)

در این مدل بعضی از z_i هاو v_j ها مساوی صفر هستند. اگر مسئله (۶) متناظر Q بصورت زیر باشد:

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

(۸)

در این مدل علاوه بر z_i و v_j هایی که در (۷) مساوی

صفر هستند، ممکن است z_i و v_j های دیگری نیز صفر باشند، به همین دلیل فضای شدنی (۸) زیر مجموعه ای از فضای شدنی (۷) است؛ پس می توان نتیجه گرفت اگر (۷) نشدنی باشد، (۸) نیز نشدنی است؛ یعنی Q نیز یک جواب نشدنی TS خواهد بود.

قضیه ۴: اگر مسئلهٔ (۵)، برای یک جواب TS مانند P نشدنی باشد؛ هر جواب دیگری مانند Q که اجزاء عدد یک P در آن نیز عدد یک باشد، نیز نشدنی است.

اثبات: فرض کنیم مسئلهٔ (۵) متناظر P بصورت زیر باشد:

$$\text{Max} \quad c_1x + d_1y©$$

$$\text{s.t}: \begin{cases} A_1x + A_2y© + u© = b \\ x, y©, u© \geq 0 \end{cases}$$

(۹)

در مدل (۹) بعضی از مولفه های بردارهای y و u مساوی صفر هستند. اگر مسئلهٔ (۵) متناظر Q بصورت زیر باشد:

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

$$\text{s.t}: \begin{cases} A_1x + A_2y" + u" = b \\ x, y", u" \geq 0 \end{cases}$$

(۱۰)

آنگاه در این مدل علاوه بر u_i و y_j هاییکه در مدل (۹) مساوی صفر هستند ممکن است u_i و y_j های دیگری نیز مساوی صفر باشند؛ به همین دلیل فضای شدنی مدل (۱۰) زیر مجموعهٔ فضای شدنی مدل (۹) خواهد بود؛ بنابر این اگر مدل (۹) نشدنی باشد مدل (۱۰) نیز نشدنی است؛ یعنی Q نیز یک جواب نشدنی TS است.

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

قضیه ۵: اگر P یک جواب شدنی TS باشد و یک جزء آن از یک به صفر تغییر کند، مسئلهٔ (۵) متناظر جواب جدید نیز شدنی خواهد بود. و مقدار بهینهٔ تابع هدف آن بدتر نخواهد شد.(یا بهتر می شود یا تغییر نمی کند.)

اثبات: با تغییر یک جزء P از یک به صفر، طبق قواعد تعریف شده یکی از متغیرهای y یا u که در مسئله (۵) مساری صفر قرار داده شده بود، می تواند بزرگتر یا مساوی صفر بشود، که موجب گسترش فضای شدنی مسئله (۵) می شود، در نتیجه چون مسئله (۵) قبلا" شدنی بود اکنون نیز شدنی خواهد بود. از آنجاییکه فضای شدنی گسترش پیدا می کند بدتر نشدن مقدار بهینهٔ تابع هدف مسئله نیز بدیهی است.

قضیه ۶: اگر P یک جواب شدنی TS باشد و یک جزء آن از صفر به یک تغییر کند، مسئلهٔ (۶) متناظر جواب جدید نیز شدنی خواهد بود.

اثبات: با تغییر یک جزء P از صفر به یک، طبق قواعد تعریف شده یکی از متغیرهای z یا v که در مسئله (۶) مساری صفر قرار داده شده بود، می تواند بزرگتر یا مساوی صفر بشود، که موجب گسترش فضای شدنی مسئله (۶) می شود، در نتیجه چون مسئله (۶) قبلا" شدنی بود اکنون نیز شدنی خواهد بود.

نکتهٔ قابل ذکر دیگر در اینجا تعریف حرکت های الگوریتم است. حرکت هایی که در این الگوریتم برای رفتن از یک جواب به جواب دیگر استفاده می شود بصورت زیر تعریف می شود.

تعریف ۳: اگر P یک جواب TS باشد، حرکت جابجایی در P عبارت است از تغییر مقدار دو جزء از P که یکی صفر و دیگری یک است.

تعریف ۴: تغییر مقدار یک جزء جواب شدنی TS از یک به صفر یا بلعکس را یک حرکت جهشی می گوییم.

لازم بذکر است در الگوریتمی که توسعه داده می شود حرکت های ممنوع با توجه به ماهیت مسئله عبارتند از؛

الف - حرکت هایی که نتیجهٔ آنها یک جواب نشدنی TS است.

ب - حرکت هایی که ممکن است باعث کاهش مقدار تابع هدف مسئلهٔ (۵) (تابع هدف سطح یک) بشوند.(این ممنوعیت در قسمت حرکت بسمت یک نقطهٔ بهینهٔ نسبی بکار می رود.)

نحوه ساختن جواب شدنی اولیه الگوریتم TS

برای ساختن جواب اولیهٔ الگوریتم ابتدا مدل شمارهٔ (۱۱) را حل می کنیم:

$$\text{Max}\quad d_2y + c_2x_0$$
$$\begin{cases} A_2y \le b - A_1x_0 \\ y \ge 0 \end{cases}$$

(۱۱)

سپس یک جواب TS مانند P را به این صورت می سازیم که اگر u_i بزرگتر از صفر باشد، جزء iام از m جزء اول P را مساوی عدد صفر و در غیر این صورت آن را برابر یک قرار می دهیم. همچنین اگر y_j بزرگتر از صفر باشد، جزء jام از n_2 جزء آخر P را مساوی عدد صفر و در غیراین صورت آن جزء را برابر یک قرار می دهیم. بدین ترتیب P بعنوان یک جواب شدنی الگوریتم TS ساخته می شود.

گامهای الگوریتم TS برای حل BLP

گام صفر: با روشی که در بخش قبل توضیح داده شد، یک جواب شدنی اولیه مانند P برای ادامهٔ الگوریتم می سازیم. این جواب را با نام Q بعنوان بهترین جواب پیدا شده تا به حال، نگهداری می کنیم. $0 \leftarrow n$ و به گام یک می رویم .

گام یک: ۱- اگر $n \ge n_0$؛ به گام دو می رویم.(n_0 تعداد حرکات در هر مرحلهٔ صعود به نقطهٔ بهینهٔ نسبی است.)

۲ - در غیر این صورت؛ یک عدد صحیح مانند r را از بین اعداد یک تا $m + n_2$ بطور تصادفی انتخاب می کنیم.

- اگر جزء rام از جواب P صفر باشد به ۲ بر می گردیم.

- در غیر این صورت، مسئله (۶) متناظر P تغییر یافته را بررسی می کنیم.

- اگر این مسئله نشدنی باشد. حرکت ممنوع است، $n \leftarrow n+1$ بدون تغییر P به ۱ بر می گردیم.

- در غیر این صورت حرکت ممنوع نیست، $n \leftarrow n+1$، P را تغییر داده و به ۱ بر می گردیم.

گام دو : - اگر P نسبت به مرحلهٔ قبل تغییر نکرده است به گام سه می رویم.

- در غیر این صورت مسئله (۵) متناظر P را حل می کنیم، این مسئله حتما" شدنی است.

- اگر مقدار بهینهٔ تابع هدف آن بهتر از مقدار بهینهٔ تابع هدف مسئله (۵) متناظر Q باشد $P \leftarrow Q$ و به گام سه می رویم.

- در غیر این صورت بدون تغییر Q به گام سه می رویم.

گام سه : - اگر معیار توقف برقرار است، به گام چهار می رویم.

- در غیر این صورت؛ با یک حرکت جهشی که در آن یک جزء صفر P به یک تغییر می کند یا با یک حرکت جابجایی فضای جواب را تغییر داده و به گام یک می رویم.

گام چهارم : جواب مسئله (۵) متناظر Q را بعنوان جواب BLP مورد نظر با این الگوریتم معرفی می کنیم و الگوریتم پایان می یابد.

تجزیه و تحلیل نتایج محاسباتی

برای تجزیه و تحلیل نتایج محاسباتی الگوریتم توسعه داده شده، مسائل برنامه ریزی خطی را با نرم افزار LINDO حل می کنیم و یک برنامهٔ رابط به زبان C++ نوشته شده است که جوابهای بدست آمده از حل مسئله های خطی توسط LINDO را گرفته و طبق الگوریتم توسعه داده شده جواب BLP مورد نظر را جستجو می کند.

دراین بخش ابتدا TS_1 را با حرکت جهشی و TS_2 را با حرکت جابجایی برای خروج از نقطهٔ بهینهٔ نسبی مورد بررسی قرار می دهیم و با حل مسائل مختلف اثر n_0 بر کارآیی محاسباتی هر دو الگوریتم را بررسی می کنیم؛ سپس کارآیی محاسباتی TS_1 و TS_2 را با هم مقایسه می کنیم تا TS مناسب تعیین شود. در انتها با حل مسائل مختلف این روش را با روش GA ارایه شده توسط ماتیو و همکارانش [۱] مقایسه می کنیم.

بررسی اثر مقدار n_0 بر کارآیی محاسباتی TS_1 و TS_2

برای بررسی اثر مقدار n_0 بر کارآیی محاسباتی هر دو الگوریتم، چهار گروه مسئله با اندازه های مختلف حل شده است.(هر گروه شامل هفت مسئله است.) متوسط زمان رسیدن به جواب مورد نظر در هر گروه برای TS_1 در جدول شمارهٔ (۱) وبرای TS_2 در جدول شمارهٔ (۲) ارایه شده است.

جدول ۱: متوسط زمان رسیدن به جواب مورد نظر با استفاده از TS_1

n_0=	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲
۷-۴-۵	۳/۹۶	۲/۹۲	۱/۸۰	۲/۳۴	۳/۰۶	۲/۸۷	۲/۴۵	۲/۷۱	۴/۲۰	-	-	-
۱۰-۶-۹	۱۱/۷	۱۹/۰۴	۹/۱۹	۱۲/۹۸	۱۲/۵۹	۱۶/۰۶	۱۹/۱۲	۱۲/۲۷	۱۸/۵۷	۱۶/۶۱	-	-
۱۵-۸-۱۰	-	-	۱۶/۱۰	۱۶/۱۵	۷/۵۳	۱۲/۳۹	۸/۵۴	۲۱/۱۵	۱۴/۴۵	۱۶/۳۵	-	-
۱۷-۷-۱۲	-	-	-	-	۷۵/۱۵	۴۰/۷۰	۵۱/۵۴	۲۹/۰۱	۵۵/۷۴	۲۱/۱۳	۷۵/۱۳	۵۸/۹۴

جدول ۲: متوسط زمان رسیدن به جواب مورد نظر با استفاده از TS_2 .

n_0=	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳
۷-۴-۵	۲/۱	۱/۷۹	۱/۲۸	۱/۶۷	۲/۶۴	۲/۲۷	۳/۲۱	۲/۴۹	۳/۳۰	۳/۸۲	-	-	-
۱۰-۶-۹	-	۱۷/۹	۱۱/۹	۱۶/۸۴	۱۴/۲۱	۹/۲۹	۱۰/۰۷	۱۳/۳۵	۱۷/۴۵	۱۵/۳۷	-	-	-
۱۵-۸-۱۰	-	-	-	۲۸/۳۲	۴۵/۴۳	۳۳/۹۰	۱۶/۵	۱۲/۴۶	۲۰/۵۲	۹/۱۲	۱۸/۰۶	۱۲/۳۸	۱۴/۶
۱۷-۷-۱۲	-	-	-	-	-	۶۵/۵۳	۸۵/۹۳	۵۷/۲۰	۶۵/۳۴	۶۹/۴۴	۵۷/۴۵	۳۵/۰۲	۴۶/۵

با توجه به جدول شمارهٔ (۱) بهترین مقدار n_0 در چهار گروه از مسائل مورد نظر بترتیب ۵،۳،۳ و ۱۲ است. همانطور که پیداست با بزرگ شدن اندازهٔ مسئله مقدار مناسب n_0 نیز بزرگتر می شود، اما رابطهٔ مشخصی نمی توان بین اندازهٔ مسئله و مقدار مناسب n_0 برقرار کرد. همچنین با توجه به جدول شمارهٔ (۲) بهترین مقدار n_0 در چهار گروه از مسائل مورد نظر بترتیب ۱۰،۶،۳ و ۱۲ است. که می توان این مقادیر را از رابطهٔ $[(m + n_2)/2 - 2.5]$ بدست آورد.

مقایسه TS_1 و TS_2

برای مقایسهٔ TS_1 و TS_2، بهترین حالت حل مسائل قسمت قبل بطور خلاصه در جدول شمارهٔ (۳) ارایه شده است. با توجه به اطلاعات این جدول در مسائل گروه دوم، سوم و چهارم TS_1 کارآیی بهتری داشته است و فقط در مسائل گروه اول TS_2 در متوسط زمان کمتری به جواب رسیده است، بنابراین TS_1 بعنوان الگوریتم TS مورد نظر پذیرفته می شود.

مقایسه روش GA ارایه شده توسط ماتیو و همکارانش با روش TS پیشنهادی

ماتیو و همکارانش [۱] روشی بر پایه الگوریتم ژنی برای حل BLP ارایه کرده اند. در این روش هر کروموزوم یک رشته اعداد حقیقی بطول n_1+n_2 است. n_1 عدد نخستین این رشته، مقادیر متغیرهای تحت کنترل تصمیم گیرندهٔ رهبر و n_2 عدد باقیمانده مقادیر متغیرهای تحت کنترل تصمیم گیرندهٔ پیرو هستند. و درواقع هر کروموزوم شدنی در این روش یک جواب قابل دستیابی BLP است؛ اما این جواب لزوما" یک نقطه راسی فضای قابل دستیابی نیست.

جدول ۳: متوسط زمان رسیدن به جواب با استفاده از TS_1 و TS_2 .

$m-n_1-n_2$	t_{TS1}	$n0_{TS1}$	t_{TS2}	$n0_{TS2}$
۷-۴-۵	۱/۸۰	۳	۱/۲۸	۳
۱۰-۶-۹	۹/۱۹	۳	۹/۲۹	۶
۱۵-۸-۱۰	۷/۵۳	۵	۹/۱۲	۱۰
۱۷-۷-۱۲	۲۱/۱۳	۱۰	۳۵/۰۲	۱۲

در الگوریتم GA ارایه شده دو روش برای تولید جمعیت نسل جدید استفاده شده است. روش اول تولید تصادفی مقدار متغیرهای تصمیم گیرندهٔ رهبر و بدست آوردن متغیرهای تصمیم گیرندهٔ پیرو با استفاده از حل مسئله 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 مسئله سطح دوم پس از اعمال تغییرات بدست می آید. در فرایند جهش این الگوریتم حدود تغییر اجزاء در هر مرحله تغییر می کند و بتدریج کاهش پیدا می کند وقتی این حدود تغییر از مقدار کوچکی مانند ε کمتر شود

جدول ۴: مقایسهٔ روش پیشنهادی با روش ماتیووهمکارانش.

شمارهٔ مسائل	$m-n_1-n_2$	t_{TS}	Δ_{TS}	t_{GA}	Δ_{GA}
۱	۶-۱-۳	۰/۵۵	۰/۰۰	۳/۵۱	۲۲/۸۰
۲	۵-۴-۵	۱/۳۰	۰/۰۰	۱۲/۵۹	۳۳/۷۴
۳	۷-۴-۵	۳/۱۶	۰/۰۰	۱۵/۴۰	۴۲/۱۶
۴	۸-۵-۶	۴/۵۳	۰/۰۰	۳۳/۱۰	۵۰/۱۲
۵	۹-۶-۸	۵/۹۰	۰/۰۰	۲۵/۷۰	۵۰/۰۶
۶	۱۰-۸-۵	۱۵/۲۵	۰/۰۰	۶۷/۶۳	۸۰/۴۷
۷	۱۱-۷-۱۰	۱۸/۳۸	۰/۰۰	-	-
۸	۱۵-۱۰-۱۰	۲۸/۳۳	۰/۳۴	-	-
۹	۲۵-۱۵-۱۵	۱۵۳/۶۰	۷/۵۳	-	-
۱۰	۳۰-۱۵-۲۰	۱۸۷/۵۹	۸/۶۱	-	-

الگوریتم متوقف شده و بهترین جواب بدست آمده را بعنوان جواب BLP گزارش می کند.

در این مقاله مقایسات با دو عامل زمان انجام محاسبات برای رسیدن به جواب مورد نظر (واحد ثانیه) و کیفیت جواب بدست آمده که بصورت زیر سنجیده می شود انجام می شود:

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

بطوریکه F^* مقدار بهینه تابع هدف سطح یک و F مقدار بدست آمده با استفاده از الگوریتمهای TS و GA برای تابع هدف سطح یک است. هر چقدر Δ برای یک جواب کمتر باشد آن جواب بهتر است.

برای مقایسهٔ دو الگوریتم ده گروه از مسائل با اندازه های مختلف که در هر گروه پنج مسئله قرار دارد حل شده است و نتایج بدست آمده در جدول شماره (۴) ارایه شده است. با توجه به دادههای جدول شمارهٔ (۴) در هفت گروه اول از مسائل Δ_{TS} برای روش TS پیشنهادی صفر است یعنی روش پیشنهادی جواب بهینه را بدست آورده است و در سه گروه آخر نیز Δ_{TS} کوچکتر یا مساوی ۸/۶۱٪ است.

در حالیکه برای روش GA در شش گروه اول از مسائل Δ_{GA} حداقل ۲۲/۸٪ است و در چهار گروه آخر، الگوریتم GA بترتیب در زمانهای ۲۰۰، ۲۵۰، ۳۰۰ و ۵۰۰ ثانیه هیچ جوابی بدست نیاورده است.

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

بنابر این روش TS با توجه به مسائل حل شده هم از نظر زمان رسیدن به جواب وهم از نظر کیفیت جواب بدست آمده برتری مشخصی نسبت به روش GA ارایه شده توسط ماتیو وهمکارانش دارد.

خلاصه و نتیجه گیری

روشهای فراابتکاری برای کاهش پیچیدگی محاسباتی ارایه شده اند و می توانند در حل BLP که یک مسئلهٔ Np_hard است مؤثر باشند.

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

پس از تخمین مقدار مناسب پارامترهای الگوریتم

توسعه داده شده با حل مسائل متعدد، ایـن روش بـا روش GA ارایه شده توسط ماتیو و همکارانش [۱] مقایسه شـده اسـت. نتـایج محاسـبات نشـان مـی دهـد کـه روش TS پیشنهادی هم از نظر زمان محاسبات و هم از نظر کیفیـت جواب بدست آمده برتری زیادی نسـبت بـه روش مـاتیو و همکارانش دارد.

در زمینهٔ استفاده از روشـهای فراابتکـاری بـرای حـل BLP مؤلفیـن ایـن مقالـه تحقیقـاتی را انجـام داده انـد، که عـــــلاوه بر مقالهٔ حاضر مقاله ای تحت عنوان حـل برنامه ریزی دوسطحی با الگوریتم ژنی آماده کرده اند که در دست چاپ می باشـد؛ همچنیـن تحقیقـات آتـی بـرای بکار گیری روشهای فراابتکاری دیگر و تعیین بهترین روش فراابتکاری برای حل BLP ادامه دارد.

قدردانی

از داوران محترم این مقاله که نظـرات ارزنـدهٔ آنـها باعث بهبود کیفیت مقاله شده است، تشـکر و سپاسـگزاری می کنیم.

مراجع

1 - Mathiue, R., Pittard, L. and Anandalingam, G. (1994). , "Genetic algorithm based approach to bilevel linear programming." *Recherche Op rationnelle / Operations Research,* Vol. 28, No. 1, PP.1-21.

2 - Falk, J. E. (1973). "A linear max-min problem." *Math. Prog.,* Vol. 5, PP. 169-188 .

3 - Bialas, W. F. and Karwan, M. H. (1982). "On two-level optimization." *IEEE Trans. Autom. Control,* Vol. 27, No. 1, PP. 11-214.

4 - Bialas, W. F. and Karwan, M. H. (1983). "Two-level linear programming." *Mgmt. Sci.,* Vol. 30, PP. 1004 – 1020.

5 - Bard, J. F. (1983). "An efficient point algorithm for a linear two-stage optimization problem." *Ops. Res.,* Vol. 38, PP. 670-684.

6 - nl , G. (1987). "A linear bilevel programming algorithm based on bicritria programming." *Comput. Ops. Res.* , Vol. 14, PP. 173-179.

7 - Candler, W. (1983). "A linear bilevel programming algorithm : a comment." *Comput. Ops. Res.,* Vol. 15, No. 3, PP. 297-298.

8 - Clark, P. A. and Westerberg, A. W. (1988). "A note on the optimality conditions for the bilevel programming problem." *Nav. Res. Logist. Q.* , Vol. 35, 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.

10 - Ben_Ayed, O. and Blair, C. E. (1990). "Computational difficulties of bilevel linear programming." *Ops. Res.,* Vol. 38, PP. 556-560.

11 - Marcotte, P. and Savard, G. (1991). "A not on the pareto optimality of solutions to the linear bilevel programming problem." *Comput. Ops. Res.,* Vol. 18, No. 4, PP. 355-359.

12 - Fortuny_Amat, J. and McCarl, B. (1981). "A representation and economic interpretation of a two-level programming problem." *J. Opl. Res. Soc.,* Vol. 32, PP. 783-792.

13 - Bard, J. F. and Moore, J. T., (1990). "A branch and bound algorithm for the bilevel programming problem." *SIAM J. Sci. Stat. Comput.,* Vol. 11, No. 2, PP. 281-292.

14 - White, D. J. and Anandalingam, G. (1973). "A penalty function approach for solving bilevel linear programs." *J. Glob. Optimiz.*, Vol. 3, PP. 397-419.

15 - Anandalingam, G. and White, D. J. (1990). "A solution for the linear static stackelberg problem using penalty functions." *IEEE Trans. Autom. Control,* Vol. 35, PP. 1170-1173.

16 - Shih, S. H., Lai, Y. J. and Lee, E. S. (1983). "Fuzzy approach for multi-level programming problem." *Comput. Ops. Res.*, Vol. 23, No. 1, PP. 773-791.

17 - Sakava, M., Nishizaki, I. and Uemura, Y. (1997). "Interactive fuzzy programming for multilevel linear programming problem." *Comput. Math. Applic.*, Vol. 6, PP. 71-86.

18 - Sahin, K. H. and Cirit, A. R. (1998). "A dual temperature simulated annealing approach for solving bilevel programming problems." *Computers Chemical Engineering,* Vol. 23, PP. 11-25.

19 - Hejazi, S. R., Memariani, A., Jahanshahloo, G. and Sepehri, M. M. (1999). "Bilevel programming solution by genetic algorithm."(Communicated).

واژه های انگلیسی به ترتیب استفاده در متن

1 - Bilevel Programming
2 - Tabu Search (TS)
3 - Computational Efficiency
4 - Modern Heuristic Method
5 - Vertex Points
6 - Accessible Region
7 - Vertex Enumeration
8 - Fuzzy Approach
9 - Branch and Bound
10 - Kth_Best
11 - Local Optima
12 - Grid Search Algorithm (GSA)
13 - Bicriteria Programming (BCP)
14 - Complementary Constraints
15 - Parametric Complementary Pivot
16 - Linear Complementary Problem (LCP)
17 - Penalty Function
18 - Noncooperative Principle
19 - Genetic Algorithm
20 - Chromosome
21 - Markov Chains