درس ۳۴

پیاده سازی الگوریتم تجزیه بندرز

تهیه شده توسط گروه آموزشی بهینه‌یاب

مقدمه

در این درس به دنبال پیاده سازی الگوریتم تجزیه بندرز در برنامه ویژال بیسیک هستیم. در ابتدا مروری بر الگوریتم تجزیه بندرز می کنیم. سپس یک مدل بهینه سازی را با استفاده از این الگوریتم حل می کنیم و در نهایت این مدل را با استفاده از زبان ویژال بیسیک با الگوریتم بندرز برنامه نویسی می کنیم.

شرح کلی روش بندرز

این روش را ریاضی دان آلمانی، ژاکوب بندرز، برای حل مسائل برنامه ریزی مختلط *MIP* طراحی کرده است.

در ابتدا مسئله *MIP* به دو زیر مسئله تقسیم می شود و سپس تبادل پاسخ های بین این دو زیر مسئله تا زمان رسیدن به پاسخ نهایی مسئله اصلی صورت می گیرد. فرض کنید مسئله *MIP* زیر داده شده است که در آن متغیرهای x متغیرهای پیوسته غیرمنفی و متغیرهای y متغیرهای عدد صحیح غیرمنفی هستند.

$$MIP(I):\begin{cases} Min\ Z = c_1x + c_2y \\ s.t. \\ A_1x + A_2y \geq b \\ x, y \geq 0, y \in \text{int} \end{cases}$$

حال فرض کنیم متغیرهای عدد صحیح y در یک مقدار صحیح غیر منفی ثابت شوند، در این صورت عبارت مربوط به y در تابع هدف زیر به یک مقدار ثابت تبدیل می شود و از تابع هدف حذف شدنی است و به مدل خطی زیر می رسیم:

$$LP(II):\begin{cases} Min\ Z = c_1x \\ s.t. \\ A_1x \geq b - A_2y \\ x \geq 0 \end{cases}$$

چون مدل *II* خطی است، پس برای آن همزاد قابل تعریف است که اگر متغیرهای دوگان آن را u فرض کنیم، مسئله همزاد متناظر با آن به صورت زیر در می آید:

$$DL(III):\{Max\ u(b-A_2y)$$
$$s.t.$$
$$\boxed{\begin{array}{l} uA_1 \le c_1 \\ u \ge 0 \end{array}} \rightarrow U$$

در مسئله *DL(III)* اگر فضای پاسخ را *U* در نظر بگیریم:

۱- *U* مستقل از متغیر *y* است.

۲- اگر *U* تهی باشد، مسئله *DL(III)* دارای پاسخ موجه نیست. آنگاه طبق نظریه همزادی مسئله *LP(II)* یا ناموجه است و یا بیکران. ولی اگر مسئله *LP(II)* بیکران باشد، به این معنی است که مسئله اصلی *MIP(I)* بیکران است اما چون در مسائل عملی و واقعی *MIP(I)* بیکران نیست، پس فرض زیر را در مورد این مسئله خواهیم داشت:

فرض ۱: فضای پاسخ *U* تهی نیست و مسئله *DL(III)* موجه است.

۳- چون مسئله *DL(III)* مستقل از *y* است و طبق فرض (۱) غیرتهی است، پس حداکثر مقدار تابع هدف یعنی *maxu(b-A₂y)* یا روی نقاط گوشه *U* اتفاق می افتد و یا در امتداد یکی از شعاع های حدی یا *extreme rays* فضای *U* افزایش نامحدود می یابند. برای جلوگیری از بیکران شدن پاسخ این مسئله فرض زیر را اضافه می کنیم.

فرض ۲: فضای پاسخ *U* محدود است و مسئله *DL(III)* دارای پاسخ بهینه محدود است و بیکران نیست، پس دارای شعاع حدی نیست.

البته حتی اگر در مسئله *DL(III)* فضای پاسخ بیکران بود نیز روش بندرز استفاده می شود زیرا برای تحقق فرض (۲) می توان محدودیتی بدیهی نظیر مجموع تمام متغیرها کوچکتر یا مساوی یک مقدار بزرگ به صورت به مسئله اضافه شود.

$$\sum u_i \le M$$

با توجه به فرضیات (۱) و (۲) می توان گفت پاسخ بهینه محدود *DL(III)* روی نقاط گوشه *U* خواهد بود و این مسئله را می توان به شکل زیر نیز نوشت:

$$DL(IV): \begin{cases} Max\ u^p (b - A_2 y) \\ u^p \in \{\text{Corner point of } U\} \\ p = 1, 2, \ldots, P \end{cases}$$

کار ما بدین صورت خواهد بود که تمام نقاط گوشه مسئله *DL(IV)* را شمارش کرده، از بین آن ها بهترین را انتخاب کنیم، اما شمارش تمام نقاط گوشه بسیار وقت گیر است و تعداد محدودیت های جبری برای تولید آن می توان بی شمار شود. پس باید تمهیدی اندیشید تا با کمترین بررسی در نقاط گوشه آن به پاسخ بهینه رسید.

از نظریه دوگانگی در برنامه ریزی خطی می دانیم که طبق خاصیت ضعیف دوگانگی داریم:

$$\underset{LP(II)}{c_1 x} \geq \underbrace{\max u^p (b - A_2 y)}_{DL(IV)} \xrightarrow{+c_2 y} c_1 x + c_2 y \geq c_2 y + \max_{u^p} u^p (b - A_2 y)$$
$$\longrightarrow Z \geq c_2 y + \max_{u^p} u^p (b - A_2 y)$$

در روابط بالا، *c_2y* را که بیشتر کم کرده، به دو طرف نامساوی اضافه کردیم و در نهایت نتیجه می گیریم مقدار تابع هدف مسئله اصلی *MIP(I)* دارای یک حد پایینی است که به ازای شمارش نقاط گوشه فضای *U* به دست می آید. پس می توان مسئله *MIP(I)* را به صورت زیر نوشت:

$$IP': \begin{cases} Min\ Z \\ s.t. \\ Z \geq c_2 y + u^p (b - A_2 y) \\ \begin{cases} p = 1, 2, \ldots, P \\ p \in P \\ y \in \text{int}, y \geq 0 \end{cases} \end{cases}$$

مسئله *IP'* یک مسئله عدد صحیح خالص است. به عبارت دیگر به جای حل مسئله *MIP(I)* که یک مسئله عدد صحیح مخلوط بود، یک مسئله را که عدد صحیح خالص است، حل می کنیم. البته در عمل به جای شمارش کامل تمام نقاط گوشه مسئله *DL(IV)* که هر کدام یک نا معادله را به مسئله *IP'* اضافه می کنند، امیدواریم با تنها یک تعداد از این نقاط گوشه بتوانیم پاسخ بهینه مسئله اصلی *MIP(I)* را تولید کنیم.

گام های الگوریتم بندرز

گام ۰: فرض کنید مسئله MIP زیر داده شده است که در آن متغیرهـای x متغیرهـای پیوسـته غیرمنفـی متغیرهـای y متغیرهای عدد صحیح غیرمنفی هستند:

$$MIP:\begin{cases} Min\ Z=c_1x+c_2y \\ s.t. \\ A_1x+A_2y\geq b \\ x,y\geq 0, y\in \text{int} \end{cases}$$

گام شروع: با یک پاسخ موجه دلخواه $u=u_0$ (که می تواند ضرورتا گوشه نیز نباشد) برای فضای موجه زیر شروع کنید.

$$\begin{cases} uA_1\leq c_1 \\ u\geq 0 \end{cases}$$

گام تکراری k ام. مسئله (IP) زیر را با افزودن نامعادله جدید مربوط به u^k حل کنید:

$$IP:\begin{cases} Min\ Z \\ s.t. \\ Z\geq c_2y+u^k\left(b-A_2y\right) \quad k=0,1,2,... \\ y\in \text{int}, y\geq 0 \end{cases}$$

پاسخ آن را z^k و y^k بنامید.

سپس مسئله (DL) زیر را به ازای $y=y^k$ حل کنید.

$$(DL)\begin{cases} Max\ w=u\left(b-A_2y\right) \\ s.t. \\ uA_1\leq c_1 \\ u\geq 0 \end{cases}$$

پاسخ بهینه آن را u^{k+1} و w^{k+1} بنامید.

گام توقف: اگر $z^k-c_2y^k\geq u^{k+1}(b-A_2y^k)$ شد، متوقف شوید در غیر این صورت $k\rightarrow k+1$ و به گام تکراری برگردید.

هنگام حل مسئله (DL) پاسخ بهینه آن یعنی w^{k+1} یک حد فوقانی برای پاسخ بهینه مسئله اصلی MIP تولید می کند زیرا $w^{k+1}+c_2y^k$ به ازای یک مقدار ثابت $y=y^k$ به دست آمده و محدودتر از مسئله اصلی MIP است. از طرف دیگر پاسخ مسئله (IP) مربوطه تنها به ازای تعداد محدودی از نقاط گوشه به دست می آید و یک نوع آزاد سازی مسئله اصلی هست، یعنی z^k یک نوع حد پایینی برای مسئله MIP است. در واقع شرط توقف هنگامی برقرار است که حد پایینی و حد بالایی به هم برسند و یا به عبارت دیگر این دو مساوی هم شوند.

حل یک مثال

با استفاده از الگوریتم بندرز مسئله زیر را حل کنید.

$$MIP:\begin{cases} Min\ Z=5x+2y_1+2y_2 \\ s.t. \\ x+3y_1+2y_2\geq 5 \\ 4x-y_1+y_2\geq 7 \\ 2x+y_1-y_2\geq 4 \\ x\geq 0, y_1,y_2\geq 0, \text{int} \end{cases}$$

در این مثال زیرمسائل تولید شده در روش بندرز به صورت زیر است:

$$LP(II):\begin{cases} Min\ Z=5x \\ s.t. \\ x\geq 5-3y_1-2y_2 \\ 4x\geq 7+y_1-y_2 \\ 2x\geq 4-y_1+y_2 \\ x\geq 0 \end{cases}$$

$$DLP(III)\begin{cases} Max\ u\left(b-A_2y\right) \\ s.t. \\ uA_1\leq c_1 \\ u\geq 0 \end{cases} \rightarrow \begin{cases} Max\ u_1\left(5-3y_1-2y_2\right)+u_2\left(7+y_1-y_2\right)+u_3\left(4-y_1+y_2\right) \\ s.t. \\ \boxed{\begin{matrix} u_1+4u_2+2u_2\leq 5 \\ u\geq 0 \end{matrix}} \rightarrow U \end{cases}$$

اکنون سعی می کنیم به جای شمارش تمام نقاط گوشه U، ان ها را یکی یکی شمارش و وارد مسئله کنیم و به ازای هر یک از آن ها نامعادله ای را در مسئله IP اضافه نماییم. حتی می توان به جای نقطه گوشه از یک پاسخ موجه فضای U نیز شروع کرد.

ابتدا با یک پاسخ موجه گوشه نظیر $u^{(0)}=(0,0,0)$ برای فضای پاسخ زیر شروع می کنیم.

$$\begin{cases} u_1 + 4u_2 + 2u_3 \le 5 \\ u \ge 0 \end{cases}$$

داریم:

$$\begin{cases} Min\ Z \\ Z \ge c_2 y + u^{(0)}(b - A_2 y) \\ y \ge 0, \text{int} \end{cases} \xrightarrow{u^{(0)}=(0,0,0)} \begin{cases} Min\ Z \\ Z \ge 2y_1 + 2y_2 \Rightarrow y^{(0)} = (0,0), Z^{(0)} = 0 \\ y_1, y_2 \ge 0, \text{int} \end{cases}$$

$$y^{(0)} = (0,0) \Rightarrow \begin{cases} w^{(1)} = Max\ 5u_1 + 7u_2 + 4u_3 \\ s.t. \\ u_1 + 4u_2 + 2u_3 \le 5 \\ u \ge 0 \end{cases} \Rightarrow w^{(1)} = 25, u^{(1)} = (5,0,0)$$

حال به بررسی شرط توقف می پردازیم:

$$w^{(1)} = 25 > Z^{(0)} - c_2 y^{(0)} = 0 - (2,2)\begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0 \Rightarrow \text{Go forward}$$

$$u^{(1)} = (5,0,0) \Rightarrow \begin{cases} Min\ Z \\ Z \ge 2y_1 + 2y_2 \\ Z \ge 25 - 13y_1 - 8y_2 \\ y_1, y_2 \ge 0, \text{int} \end{cases} \Rightarrow Z^{(1)} = 4, y^{(1)} = (2,0)$$

$$y^{(1)} = (1,1) \Rightarrow \begin{cases} w^{(2)} = Max\ -u_1 + 9u_2 + 2u_3 \\ s.t. \\ u_1 + 4u_2 + 2u_3 \le 5 \\ u_i \ge 0, i = 1,2,3 \end{cases} \Rightarrow w^{(2)} = 11.25, u^{(2)} = (0, 1.25, 0)$$

حال به بررسی شرط توقف می پردازیم:

$$w^{(2)}=10>Z^{(1)}-c_2y^{(1)}=4-(2,2)\begin{pmatrix}2\\0\end{pmatrix}=0\Rightarrow \text{Go forward}$$

$$u^{(2)}=(0,1.25,0)\Rightarrow\begin{cases}Min\ Z\\ Z\ge 2y_1+2y_2\\ Z\ge 25-13y_1-8y_2\\ Z\ge 8.75+3.25y_1+0.75y_2\\ y_1,y_2\ge 0,\text{int}\end{cases}\Rightarrow Z^{(2)}=10.25,y^{(2)}=(0,2)$$

$$y^{(2)}=(0,2)\Rightarrow\begin{cases}w^{(3)}=Max\ u_1+5u_2+6u_3\\ s.t.\\ u_1+4u_2+2u_3\le 5\\ u_i\ge 0, i=1,2,3\end{cases}\Rightarrow w^{(3)}=15,u^{(3)}=(0,0,2.5)$$

حال به بررسی شرط توقف می پردازیم:

$$w^{(3)}=15>Z^{(2)}-c_2y^{(2)}=10.25-(2,2)\begin{pmatrix}0\\2\end{pmatrix}=6.25\Rightarrow \text{Go forward}$$

$$u^{(3)}=(0,0,2.5)\Rightarrow\begin{cases}Min\ Z\\ Z\ge 2y_1+2y_2\\ Z\ge 25-13y_1-8y_2\\ Z\ge 8.75+3.25y_1+0.75y_2\\ Z\ge 10-0.5y_1+4.5y_2\\ y_1,y_2\ge 0,\text{int}\end{cases}\Rightarrow Z^{(3)}=12,y^{(3)}=(1,0)$$

$$y^{(3)}=(1,0)\Rightarrow\begin{cases}w^{(4)}=Max\ 2u_1+8u_2+3u_3\\ s.t.\\ u_1+4u_2+2u_3\le 5\\ u_i\ge 0, i=1,2,3\end{cases}\Rightarrow w^{(4)}=10,u^{(4)}=(0,1.25,0)$$

حال به بررسی شرط توقف می پردازیم:

$$w^{(4)}=10=Z^{(3)}-c_2y^{(3)}=12-(2,2)\begin{pmatrix}1\\0\end{pmatrix}=10\Rightarrow \text{stop}\square.$$

نکته: در هر تکرار، مقدار $z^{(k-1)}$ حد پایین و مقدار $w^{(k)}+c_2y^{(k-1)}$ حد بالا برای جواب بهینه مسئله اصلی است.

پیاده سازی الگوریتم تجزیه بندرز

در این بخش به پیاده سازی الگوریتم تجزیه بندرز با استفاده از نرم افزار ویژال بیسیک می پردازیم. قسمت های مختلف این برنامه در ادامه با جزییات بیان می شود.

در این بخش دو مدل *DLP* و *IP* و محدودیت های مربوط به آن ها تعریف می شود.	`Dim cplexDLP As New Cplex()`
	`Dim cplexIP As New Cplex()`
	`Dim rngIP(1)() As IRange`
	`Dim rngDLP(1)() As IRange`

متغیرهای مورد استفاده در مدل *DLP* و *IP* تعریف می شود.	`Dim z As INumVar`
	`Dim x As INumVar`
	`Dim y1 As INumVar`
	`Dim y2 As INumVar`
	`Dim u1 As INumVar`
	`Dim u2 As INumVar`
	`Dim u3 As INumVar`

مقدار جواب بهینه متغیرهای مدل *DLP* و *IP* تعریف می شود که به صورت پیوسته هستند.	`Dim u1bar As Double`
	`Dim u2bar As Double`
	`Dim u3bar As Double`
	`Dim zbar As Double`
	`Dim xbar As Double`
	`Dim y1bar As Double`
	`Dim y2bar As Double`

مقدار بهینه تابع هدف مدل های *DLP* و *IP* تعریف می شود که به صورت پیوسته فرض می شوند.	`Dim wobj As Double` `Dim zobj As Double`
فرایند الگوریتم بندرز به این صورت است که محدودیت به صورت تدریجی به مدل *IP* اضافه می شود. تعداد محدودیت های اضافه شده در این متغیر ذخیره می شود که یک متغیر گسسته است.	`Dim counterIP As Integer`
شرط توقف یک متغیر دودویی است که مقدار درست و غلط می گیرد و در اینجا تعریف می شود و مقدار اولیه ان غلط است.	`Dim stoppingcondition As Boolean` `stoppingcondition = False`

<table>
<tr><td><pre>z = cplexIP.NumVar(Double.MinValue,
Double.MaxValue, ILOG.Concert.NumVarType.Float)
x = cplexIP.NumVar(0, Double.MaxValue,
ILOG.Concert.NumVarType.Float)
y1 = cplexIP.NumVar(0, Integer.MaxValue,
ILOG.Concert.NumVarType.Int)
y2 = cplexIP.NumVar(0, Integer.MaxValue,
ILOG.Concert.NumVarType.Int)

u1 = cplexDLP.NumVar(0, Double.MaxValue,
ILOG.Concert.NumVarType.Float)
u2 = cplexDLP.NumVar(0, Double.MaxValue,
ILOG.Concert.NumVarType.Float)
u3 = cplexDLP.NumVar(0, Double.MaxValue,
ILOG.Concert.NumVarType.Float)</pre></td><td>تعریف متغیرهای مدل های DLP و IP انجام می شود و نوع آن ها(گسسته یا پیوسته بودن) و حد بالا و پایین انجام می شود.</td></tr>
<tr><td><pre>u1bar = 0
u2bar = 0
u3bar = 0
counterIP = 0
wobj = 0
zobj = 0</pre></td><td>مقادیر اولیه برای متغیر برابر صفر در نظر گرفته می شوند.</td></tr>
<tr><td><pre>Dim experssion(3) As ILOG.Concert.IIntExpr
experssion(1) = cplexDLP.Prod(u1, 0) ' (0, u1)
experssion(2) = cplexDLP.Prod(u1, 0)
experssion(3) = cplexDLP.Prod(u1, 0)</pre></td><td>برای تولید معادلات از این متغیر استفاده می شود و مقدار دهی اولیه هم انجام می شود.</td></tr>
<tr><td><pre>Dim objfunctionip As ILOG.Concert.IObjective =
cplexIP.AddMinimize()
Dim objfunctiondlp As ILOG.Concert.IObjective =
cplexDLP.AddMaximize()</pre></td><td>مدل IP به دنبال کمینه کردن تابع هدف و مدل DLP به دنبال بیشینه کردن تابع هدف است که توابع هدف مربوط به آن ها اینجا تعریف می شود.</td></tr>
<tr><td><pre>experssion(3) = cplexDLP.Sum(cplexDLP.Prod(1,
u1), cplexDLP.Prod(4, u2), cplexDLP.Prod(2, u3))

rngDLP(0) = New IRange(1) {}
rngDLP(0)(1) = cplexDLP.AddLe(experssion(3), 5,
"DLP1")</pre></td><td>محدودیت مدل DLP در طول اجرای الگوریتم ثابت است و لذا در ابتدا تعریف می شود و سپس به محدودیت های این مدل اضافه می شود.</td></tr>
<tr><td><pre>rngIP(0) = New IRange(1000) {}
experssion(1) = cplexIP.Prod(1, z)
cplexIP.AddToExpr(objfunctionip, experssion(1))</pre></td><td>تعداد محدودیت های مدل IP نامشخص است و وابسته به تعداد برش های اضافه شده در الگوریتم دارد. در این جا حداکثر تعداد این برش ها ۱۰۰۰ فرض می شود. تابع هدف مدل IP در طول اجرای این الگوریتم ثابت و برابر Z است.</td></tr>
<tr><td><pre>Do While stoppingcondition = False</pre></td><td>از این جا به بعد فرایند تکراری الگوریتم تجزیه بندرز شروع</td></tr>
</table>

	می شود. در صورتی که متغیر *stoppingcondition* برابر *False* باشد، الگوریتم ادامه می یابد. در غیر این صورت الگوریتم به جواب بهینه رسیده است.

`experssion(1) = cplexIP.Sum(cplexIP.Prod(1, z), cplexIP.Prod(-2, y1), cplexIP.Prod(-2, y2), cplexIP.Prod(3 * u1bar - u2bar + u3bar, y1), cplexIP.Prod(2 * u1bar + u2bar - u3bar, y2))`
`rngIP(0)(counterIP) = cplexIP.AddGe(experssion(1), 5 * u1bar + 7 * u2bar + 4 * u3bar, "IP" & counterIP)`
`counterIP += 1`
	محدودیت مدل *IP* با توجه به مقدار بهینه مدل *DLP* ایجاد می شود. در طول اجرای الگوریتم، متغیر *counterIP* افزایش می یابد تا محدودیت ها به تدریج به مدل *IP* اضافه شود. پس از اضافه شدن این محدودیت، یک واحد به متغیر *counterIP* اضافه می شود.

`cplexIP.Solve()`
`zbar = cplexIP.GetValue(z)`
`y1bar = cplexIP.GetValue(y1)`
`y2bar = cplexIP.GetValue(y2)`
`zobj = cplexIP.ObjValue`
	اکنون می توان مدل *IP* را حل کرد. مقدار بهینه تابع هدف در متغیر *zbar* و *zobj* و مقدار جواب متغیرهای *y1* و *y2* در متغیرهای *y1bar* و *y2bar* ذخیره می شود.

`experssion(2) = cplexDLP.Sum(cplexDLP.Prod(5 - 3 * y1bar - 2 * y2bar, u1), cplexDLP.Prod(7 + y1bar - y2bar, u2), cplexDLP.Prod(4 - y1bar + y2bar, u3))`
`objfunctiondlp.ClearExpr()`
`cplexDLP.AddToExpr(objfunctiondlp, experssion(2))`
	با توجه به جواب مدل *IP*، مدل *DLP* ساخته می شود که تنها تابع هدف این مدل تغییر می کند و محدودیت آن بدون تغییر باقی می ماند.

`cplexDLP.Solve()`
`u1bar = cplexDLP.GetValue(u1)`
`u2bar = cplexDLP.GetValue(u2)`
`u3bar = cplexDLP.GetValue(u3)`
`wobj = cplexDLP.ObjValue`
	اکنون می توان مدل *DLP* را حل کرد و مقدار بهینه تابع هدف را در *wobj* و مقدار بهینه متغیرها را در *u1bar*، *u2bar* و *u3bar* ذخیره کرد.

`If wobj <= zobj - 2 * y1bar - 2 * y2bar Then`
`stoppingcondition = True`
`End If`
 	
`Loop`
 	
`cplexDLP.End()`
`cplexIP.End()`
	شرط توقف کنترل می شود. اگر برقرار بود متغیر *stoppingcondition* برابر *True* می شود و وقتی به *Loop* می رسیم، برنامه به *while* باز می گردد و چون شرط درست است لوپ دیگر تکرار نمی شود و الگوریتم به پایان می رسد.

برای پاک کردن حافظه از دستور *end* استفاده می شود.	