دانشکده علوم ریاضی
دانشگاه صنعتی شریف

تحقیق در عملیات ۱

محمدهادی فروغمنداعرابی
پاییز ۱۳۹۹

ادامه سیمپلکس عمومی

جلسه هشتم

نگارنده: زهرا هادی زاده

در ادامه طبق همان‌طور که در کلاس مطرح می‌شود بخش‌بندی و مطالب بیاید.

۱ مروری بر الگوریتم سیمپلکس

در این قسمت به طور خلاصه الگوریتم سیمپلکس را شرح میدهیم:
فرض کنید برنامه ریزی خطی زیر به ما داده شده است:

$$\begin{aligned} \text{بیشینه کن} \quad & c^T x \\ \text{که} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

به ازای هر سطر ماتریس A یک متغیر منحصر به فرد (به سطر i ام متغیر x_{n+i} را)اضافه می کردیم تا $Ax \leq b$ به فرم $A' = b$ بدست بیاید. که در آن ماتریس A' الحاق ماتریس A است با ماتریس I_m است. در ابتدا تمام متغیر های $x_1, x_2, ..., x_n$ را با صفر و متغیر x_{n+i} را با b_i مقدار دهی اولیه میکردیم.
حال قیود را طوری تغییر میدادیم که تمام متغیر های x_{n+i} در یک طرف تساوی باشند و باقی متغیر ها در یک سمت دیگر و تابع هدف را که بر اساس متغیر های $x_1, ..., x_n$ است را z مینامیم.
به نوشتاری که حاصل شد یک تابلو میگفتیم. منظور مجموعه ی قیودی است که یک سمت آن ها یک متغیر وجود دارد و تابع هدف نیز بر اساس باقی متغیر ها تعریف شده است. و B را مجموعه تمام x_{n+i} ها تعریف کردیم. در واقع به نوشتار زیر یک تابلو میگفتیم

$$\frac{x_B = p + Qx_N}{z = z_\circ + r^T x_N}$$

حال گام لولا را بر روی تابلو آنقدر انجام میدادیم تا ضرایب تمام متغیر های آمده در تابع هدف منفی شوند. و عدد ثابتی که در تابع هدف ذکر میشود، جواب نهایی معرفی میشد.
گام لولا : در هر مرحله یک متغیر از متغیر های آمده در تابع هدف را انتخاب میکنیم (مانند متغیر x_k)که ضریب آن در تابع هدف مثبت است و به مقدار آن اضافه میکردیم و از مقدار متغیر های B کم میکردیم (به طوری که قیود برقرار باشند)، تا زمانی که دیگر نتواند افزایش پیدا کند. (اگر برنامه ریزی خطی کراندار باشد، حتما این اتفاق میافتد). و زمانی که دیگر نتواند افزایش پیدا کند معادل است با زمانی که یکی از متغیر های B، مانند x_j صفر شود و با افزایش x_k، x_j مجبور باشد کمتر از صفر شود. حال x_j را از B حذف میکردیم. و x_k را به B اضافه میکردیم و قیود مربوط به x_j را طوری تغییر میدادیم که x_j به سمت راست قید و x_k به سمت چپ قید بیاید و در باقی قید ها به جای متغیر x_k، معادل آن را قرار میدهیم و گام لولا را تکرار میکردیم.

۲ حل برنامه ریزی خطی به فرم نرمال با کمک الگوریتم سیمپلکس

در این جلسه بررسی میکنیم که چطور میشود برنامه ریزی خطی زیر را با کمک الگوریتم سیمپلکس حل کرد:

$$\begin{aligned} \text{بیشینه کن} \quad & c^T x \\ \text{که} \quad & Ax = b \\ & x \geq \circ \end{aligned}$$

ابتدا میگوییم چرا الگوریتم سیمپلکس که در جلسه پیش بیان کردیم، در نگاه اول بر روی این برنامه ریزی خطی، کار نمیکند.
زیرا در این برنامه ریزی خطی نمیتوان به سادگی یک جواب شدنی اولیه، پیدا کرد. (به یاد بیارید که در برنامه ریزی خطیی که الگوریتم سیمپلکس را روی آن اجرا کردیم به متغیر ها مقدار اولیه داده بودیم.)

بنابراین تلاش میکنیم با حل یک برنامه ریزی خطی دیگر، مقدار اولیه ای برای این برنامه ریزی خطی پیدا کنیم، و سپس الگوریتم سیمپلکس را روی آن انجام دهیم.
ابتدا هر سطر ماتریس A و بردار b را به گونه ای تغییر میدهیم تا تمام b_i ها مثبت شوند. (اگر b_i منفی بود سطر i ام ماتریس A و b_i را در -۱ ضرب میکنیم.)
حالا به هر کدام از قیود یک متغیر اضافه کنیم. به قید i ام، متغیر x_{n+i} را اضافه میکنیم و برنامه ریزی خطی زیر را در نظر میگیریم:

$$\begin{aligned} \text{بیشینه کن} \quad & -(x_{n+۱} + x_{n+۲} + \ldots + x_{n+m}) \\ \text{که} \quad & \overline{A}\overline{x} = b \\ & \overline{x} \geq \circ \end{aligned}$$

که در آن $\overline{A} := [A|I_m]$ و $\overline{x} = (x_۱, x_۲, \ldots, x_{n+m})$.
این برنامه ریزی خطی را برنامه ریزی خطی کمکی مینامیم. و برنامه ریزی خطی اولیه را برنامه ریزی خطی اصلی مینامیم.
انتظار داریم جواب بهینه برنامه ریزی خطی کمکی صفر باشد اگر و تنها اگر برنامه ریزی خطی اصلی ما دارای جواب شدنی باشد. اثبات: اگر برنامه ریزی خطی اصلی دارای جواب شدنی باشد، آنگاه مقدار متغیر های $x_۱, \ldots, x_n$ را همان مقدار جواب شدنی برنامه ریزی خطی اصلی میگیریم و مقادیر $x_{n+۱}, \ldots, x_{n+m}$ را برابر صفر در نظر میگیریم این یک جواب بهینه صفر برای برنامه ریزی خطی کمکی است زیرا تابع هدف در آن صفر است و در قیود نیز صدق میکند.
اگر برنامه ریزی خطی کمکی دارای جواب بهینه صفر باشد آنگاه تمام $x_{n+i} = \circ$. مقادیر باقی متغیر ها را برای متغیر های برنامه ریزی خطی اصلی نیز در نظر بگیریم یک جواب شدنی بدست می آید زیرا در قیود آن صدق میکند.

برای برنامه ریزی خطی کمکی میتوان یک جواب شدنی، اولیه پیدا کرد زیرا میتوان تمام متغیر های $x_۱, \ldots, x_n$ را برابر صفر قرار داد و متغیر x_{n+i} را برابر b_i قرار داد. بنابراین الگوریتم سیمپلکس بر روی برنامه ریزی خطی کمکی میتواند اجرا شود. الگوریتم سیمپلکس یک جواب شدنی برای برنامه

ریزی خطی اصلی (اولیه) میدهد. تابلویی که برنامه ریزی خطی کمکی در آخر به ما میدهد را در نظر بگیرید. متغیر های $x_{n+۱}, x_{n+۲}, ..., x_{n+m}$ را از هر یک از قیود و B حذف کنید. بیس(B) جدید را در نظر بگیرید میخواهیم با این B یک تابلو برای برنامه ریزی خطی اصلی بسازیم. قیود برنامه ریزی خطی اولیه با حذف x_{n+i} ها از قیود برنامه ریزی خطی کمکی بدست می آید بنابراین کافی است تابع هدف برنامه ریزی خطی اصلی را طوری تغییر دهیم که هیچ یک از متغیر های B در آن نباشد. و برای اینکار تمام متغیر های در تابع هدف که در B هستند را میتوان با معادل آن ها در قیود، جایگزین کرد.(زیرا هر کدام از اعضای B مانند x دقیقا یک قید وجود دارد که سمت چپ آن قید، همان عضو x است و سمت راست آن قید، معادله ای با متغیر هایی که در B نیستند، آمده است.

۳ انتخاب متغیر مناسب برای هر گام لولا

نکته دیگری که وجود دارد این است که قانون لولا را روی چه متغیری انجام دهیم؟ برای انتخاب کردن بین متغیر ها الگوریتم های مختلفی وجود دارد:

- largest coefficient: هر متغیری که ضریبش در تابع هدف بیشتر بود.
- largest increase: متغیری که زیاد کردنش بیشترین تاثیر را بر روی تابع هدف دارد.
- steepest edge: متغیری را بگیریم که جهت بردار x بعد از تغییرات و قبل از تغییر کمترین زاویه را با هم داشته باشند. به زبانی دیگر میتوان گفت متغیری را بگیریم که مقدار ضرب داخلی بردار x قبل و بعد آن بیشترین شود. بعنی مقدار زیر بیشترین شود.

$$\frac{c^T(x_{new} - x_{old})}{||x_{new} - x_{old}||}$$

- Bland's Rule: از بین متغیر هایی که در تابع هدف ضریبشان مثبت است آنی را انتخاب کنیم که اندیسش کمتر است.
- Random Edge: از بین متغیر هایی که در تابع هدف ضریبشون مثبت است تصادفی یکی را انتخاب کنیم.

قضیه: اگر الگوریتم Bland's Rule را در پیش بگیریم، هیچ گاه دو بار یک تابلوی خاص و مقدار تابع هدف یکسان، را نمیبینیم.
اثبات: فرض خلف میکنیم: فرض کنید دو بار به یک B خاص برسیم. fickle variable یا F را متغیر هایی تعریف میکنیم که در این دور یک بار از base یا B وارد و یک بار خارج میشوند. چون متغیر های F یک بار از بیس (B) خارج شده اند بنابراین مقدار آن ها زمانی که در دور، در خارج از بیس هستند، صفر است. و چون در این دور متغیر ها مقدارشان تغییر نمیکند زیرا اگر متغیر هایی که در تابع هدفمان آمده اند را زیاد کنیم مقدار تابع هدف زیاد میشود و دیگر در دور نمی افتیم. بنابراین چون مقدار تابع هدف یکسان است در یک دور، بنابراین تمام متغیر های F، مقدارشان صفر است و مقدارشان در کل دور همین است.
در بین متغیر های F، متغیر با بزرگترین اندیس را v بنامید.
حال آن لحظه ای از دور را در نظر بگیرید که متغیر v وارد base (B) میشود. چرا متغیر هایی که اندیسشان از v کوچک تر است را نمیتوانیم وارد B کنیم؟ زیرا ضرایب آن متغیر ها در تابع هدف منفی است. بنابراین داریم:

$$B = k_۱, k_۲, ..., k_m$$

$$N = L_۱, L_۲, ..., L_{n-m}$$

(N شامل متغیر هایی در برنامه ریزی خطی است که در بیس (B) نیامده اند.)

$$v = L_\beta$$

(در لحظه ای که میخواهیم v را وارد کنیم v جز متغیر های base نیست و در N وجود دارد.)
$r_\beta \geq 0$ and $r_j \leq 0$ for all j such that $L_j \in f \setminus \{v\} \setminus B$

لحظه ای که میخواهیم v را خارج کنیم، را در نظر بگیرید. داریم:

$$B' = k'_۱, k'_۲, ..., k'_m$$

$$N' = L'_۱, L'_۲, ..., L'_{n-m}$$
$$k'_{\alpha'} = v$$
$$L'_{\beta'} = u$$

که در آن u، متغیری است که در لحظه ای که v را میخواهیم خارج کنیم، آن را وارد میکنیم.

چرا نمیتوانیم متغیر های با اندیس کوچکتر را خارج کنیم؟ چون اندیس های کوچیکتر محدودیتی برای متغیر u ایجاد نمیکنند.چون مقدار u به اندازه صفر تغییر میکند(چون u در F است و مقدارش در دور صفر باقی میماند.) بنابراین ضریب u در معادله مربوط به متغیر ها با ضرایب کوچیکتر منفی است.

بنابراین:

$$q'_{\alpha'\beta'} \leq 0 \text{ and } q'_{i\beta'} \geq 0 \text{ for all i such that } k'_i \in f \setminus \{v\} \setminus N$$

حال برنامه ریزی خطی زیر را در نظر بگیرید.

$$\begin{aligned} \text{بیشینه کن} \quad & c^T x \\ \text{که} \quad & Ax = b \\ & X_{F\setminus\{v\}} \geq ۰ \\ & X_v \leq ۰ \\ & x_{N\setminus F} = ۰ \end{aligned}$$

دو ادعا زیر را برای برنامه ریزی خطی بالا ثابت میکنیم که با هم تناقض دارند. بنابراین امکان ندارد دور وجود داشته باشد.

ادعا ۱: جواب مربوط به B بهینه است. (جوابی که در برنامه ریزی خطی زمانی که وارد v میشود.)

چرا؟

الف) B یک جواب شدنی برای این برنامه ریزی خطی است. زیرا در قیود مساله اصلی که به وضوح صدق میکرد بنابراین در شروط $Ax = b$ صدق میکند. در متغیر هایی که در N هستند همواره صفر است (زیرا متغیر هایی که در بیس نیستند صفرند.) بنابراین در شرط $X_{N\ F} = ۰$ صدق میکند. و گفتیم در متغیر های F صفرند (زیرا هر متغیر در F یک بار در دور از B وارد و خارج میشود بدون اینکه تابع هدف زیاد شود و چون وقتی از B خارج میشود صفر است بنابراین همواره برابر صفر است.) بنابراین در دو شرط $X_{F\setminus\{v\}} \geq ۰$ و $X_v \leq ۰$ صدق میکند.

ب) جواب مربوط به B بهینه است: ضرایب متغیر هایی که میتوانند مثبت باشند، در تابع هدف کوچکتر مساوی صفر است. و ضرایب متغیر هایی که میتوانند منفی باشند،در تابع هدف بزرگتر مساوی صفر است. زیرا همانطور که ثابت کرده ایم:

$$r_\beta \geq 0 \text{ and } r_j \leq 0 \text{ for all j such that } L_j \in F \setminus \{v\} \setminus B$$

برای همین به تابع هدف نمیتوانیم مقداری اضافه کنیم.

ادعا ۲: جواب این برنامه ریزی خطی کران ندارد.

تابلو β' را در نظر بگیرید جواب مربوط به این تابلو یک جواب شدنی برای برنامه ریزی خطی بالا است. زیرا در قید ها صدق میکند. حال متغیر u را در این تابلو در نظر بگیرید(متغیری که زمانی که v حذف میشد، اضافه میشد). ثابت میکنیم u را میتوان هر مقداری اضافه کرد بدون تغییر متغیر های دیگر در تابع هدف و بنابراین تابع هدف میتواند، هر مقدار اضافه شود. بنابراین جواب این برنامه ریزی خطی کران ندارد.

چرا میتوان u را هر مقدار اضافه کرد ؟ ثابت میکنیم به ازای هر مقدار t میتوان u را t واحد اضافه کرد، بدون اینکه شرایط برنامه ریزی خطی برهم بخورد. ثابت کرده بودیم

$$q'_{\alpha'\beta'} \leq 0 \text{ and } q'_{i\beta'} \geq 0 \text{ for all i such that } k'_i \in f \setminus \{v\} \setminus N$$

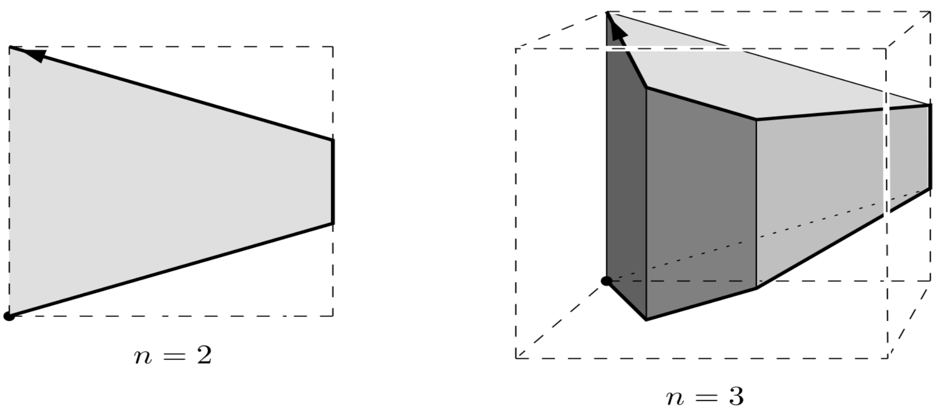
بنابراین ضریب u در هر یک از معادلات که طرف چپ آنها متغیر های غیر از v است، مثبت است بنابراین با افزایش u باید مقدار هر کدام ازین متغیر های را افزایش دهیم تا معادلات به هم نخورند. و چون در برنامه ریزی خطی این متغیر ها باید مثبت باشند، افزایش آنها مشکلی ایجاد نمیکند. و ضریب متغیر u در معادله ای که طرف چپ آن v است، منفی است. بنابراین با افزایش u مقدار v را باید کاهش دهیم تا مشکلی ایجاد نکند. و چون در برنامه ریزی خطی $X_v \leq ۰$ بنابراین کاهش این متغیر ،شرایط برنامه ریزی خطی را نگه میدارد. و متغیر های خارج از B نیز تغییری نمیکنند. بنابراین به ازای هر مقدار t میتوان u را t واحد افزایش داد، و چون باقی متغیر های خارج base تغییر نمیکنند(زیرا فقط متغیرهایی تغییر میکنند که

در سمت چپ معادلات آمده باشند بنابراین باید در base وجود داشته باشند)، بنابراین تابع هدف به اندازه ضریب u در t تغییر میکند. بنابراین تابع هدف نیز کران ندارد.

دو ادعای بالا با هم در تناقضند بنابراین امکان ندارد که برنامه ریزی خطی اولیه در دور بیافتد.

۴ آیا الگوریتم سیمپلکس سریع است؟

میدانیم الگوریتم سیمپلکس در عمل $۲m$ تا $۳m$ تا لولا میزند. ولی مثال هایی وجود دارند بسیار زمان میبرد تا به جواب برسد. در این مثال ها که یکی از آنها را توضیح میدهیم، سعی میشود الگوریتم سیمپلکس را مجبور کند تمام رئوس چندوجهی را برای پیدا کردن جواب بهینه، چک کند.

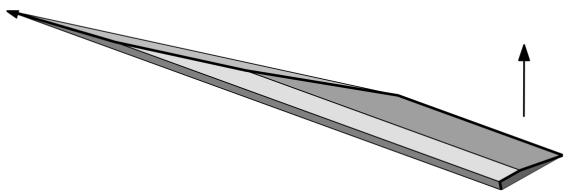


شکل ۱

به طور مثال شکل بالا اگر تابع هدف ما برابر y باشد و ما از $(۰, ۰)$ شروع کرده باشیم آنگاه یکی از راه ها برای سیمپلکس اینست که هر کدام از رئوس را چک کند، بنابراین مثالی وجود دارد که سیمپلکس لزوما در آن به خوبی $۲m$ و $۳m$ کار نمیکند. شکل ۲ ، مثال واقعی ای است که الگوریتم سیمپلکس در آن، تمام رئوس را میپیماید

با اینحال میدانیم الگوریتم سیمپلکس حداکثر $e^{c\sqrt{n \ln n}}$ لولا میزند.

حال میخواهیم کران پایینی پیدا کنیم که بگوییم الگوریتم سیمپلکس با انتخاب مناسب متغیر ها، نمیتواند از آن حد کمتر تضمین کند برابر تعداد لولا زدن ها.
اگر یک الگوریتم مثل سیمپلکس داشتیم که سعی میکرد در هر لولا بهترین انتخاب را کند، به این معنی که با آن تغییرات، در کمترین زمان به جواب بهینه برسد.ت(وجه کنید ماهیت لولا زدن را تغییر نداده ایم و فقط در مورد انتخاب متغیری برای گام لولا زمانی که نمیتوانیم تابع هدف را موضعی بیشتر کنیم، حرف زده ایم) آنگاه این الگوریتم چه مقدار سریع بود؟
میدانیم اگر نشود این الگوریتم را سریع انجام داد، الگوریتم هایی که روی یال راه میروند به طور کلی زمان اجرایی بدی خواهند داشت.
یه حدس وجود داشت که بیان میکرد بین هر دو راس یک چند وجهی که تعداد وجه هایش کم است بین هر دو راسی یک مسیر خیلی کوتاه وجود دارد. که بعدها این حدس نقض شد بعد حدسی زده شد که بیان میکرد در حدود $o(n)$ تا فاصله هست بین دور ترین راس ها در یک چندوجهی. این همچنان نه ثابت شده، نه رد شده. کرانی که فعلن ثابت شده $n^{۱+\ln n}$ است.



شکل ۲