دانشکده علوم ریاضی
دانشگاه صنعتی شریف

تحقیق در عملیات ۱

محمدهادی فروغمنداعرابی
پاییز ۱۳۹۹

کاربرد برنامه ریزی خطی در زمان بندی

جلسه شانزدهم

نگارنده: آیدا افشارمحمدیان

۱ مروری بر مباحث گذشته

در جلسه ی گذشته کاربرد برنامه ریزی خطی در اثبات برخی از قضیه های گراف را بررسی کردیم. در این جلسه میخواهیم با استفاده از برنامه ریزی خطی، یک الگوریتم تقریبی برای یک مساله ی زمان بندی ارایه دهیم.

۲ مساله ی زمان بندی

۱.۲ صورت مساله

در یک دفتر فنی m **نوع** دستگاه فتوکپی وجود دارد. اپراتور دفتر فنی باید n کار مختلف فتوکپی را با این دستگاه ها انجام دهد. میدانیم که دستگاه i ، فایل j را در چه مدت زمانی کپی می گیرد. سوال این است که کار ها را چگونه بین دستگاه ها تقسیم کنیم تا کار اپراتور دفتر فنی سریعتر تمام شود. در واقع میخواهیم فاصله ی زمانی بین شروع اولین کار تا پایان آخرین کار را مینیمم کنیم.

	Single B&W	Duplex B&W	Duplex Color
Master's thesis, 90 pages two-sided, 10 B&W copies	—	45 min	60 min
All the Best Deals flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Buyer's Paradise flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Obituary, 2 pages two-sided, 100 B&W copies	—	2 min	3 min
Party platform, 10 pages two-sided, 5,000 color copies	—	—	3h 30 min

به طور مثال در شکل بالا ۳ دستگاه و ۵ کار فتوکپی داریم. اگر کار ها را مطابق شکل زیر بین دستگاه ها تقسیم کنیم، فاصله ی زمانی بین آغاز تا پایان کار ۴ ساعت و نیم می شود.

	Single B&W	Duplex B&W	Duplex Color
Master's thesis, 90 pages two-sided, 10 B&W copies	—	45 min	60 min
All the Best Deals flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Buyer's Paradise flyer, 1 page one-sided, 10,000 B&W copies	2h 45 min	4h 10 min	5h 30 min
Obituary, 2 pages two-sided, 100 B&W copies	—	2 min	3 min
Party platform, 10 pages two-sided, 5,000 color copies	—	—	3h 30 min

چون جواب بهتر از ۵:۳۰ داریم، این‌ها در جواب بهینه نیستند!

نکته: همان طور که در شکل می بینید، تعدادی از دستگاه ها یک کار را در زمانی بیشتر از جواب شدنی ما (چهار ساعت و نیم) انجام میدهند، پس واضح است که در جواب بهینه، این کار ها، به این دستگاه ها تعلق نمی گیرند. بعدا در آرام سازی برنامه ریزی صحیح ، از این نکته استفاده می کنیم.

حل این مساله در حالت کلی سخت (NP-HARD) است. ما به دنبال ارایه ی یک الگوریتم ۲ ـ تقریب برای این سوال هستیم.

۲.۲ نماد گذاری

$$M := \{۱, ۲, ..., m\} \quad \rightarrow \quad \text{ماشین ها}$$

$$J := \{m + ۱, m + ۲, ..., m + n\} \quad \rightarrow \quad \text{کار ها}$$

$$x_{ij} = \begin{cases} ۱ & \text{ماشین i کار j را انجام دهد} \\ ۰ & \text{otherwise} \end{cases}$$

برای راحتی فرض می کنیم، مدت زمان انجام یک کار توسط یک ماشین همیشه مثبت است. اگر کاری توسط ماشینی انجام نشود، مدت زمان آن را بی نهایت در نظر میگیریم.

$$d_{ij} \quad \rightarrow \quad \text{مدت زمان انجام کار j توسط ماشین i}$$

$$d_{ij} > ۰, \qquad \forall ۱ \leq i \leq m, \quad \forall m + ۱ \leq j \leq m + n$$

۳ برنامه ریزی صحیح

$$\begin{array}{lrll} \text{Minimize} & t & & \\ \text{subject to} & \sum_{i \in M} x_{ij} & = 1 & \text{for all } j \in J \\ & \sum_{j \in J} d_{ij} x_{ij} & \leq t & \text{for all } i \in M \\ & x_{ij} & \geq 0 & \text{for all } i \in M, j \in J \\ & x_{ij} & \in \mathbb{Z} & \text{for all } i \in M, j \in J. \end{array}$$

توضیح قید ها:

- قید اول تضمین می کند که هر کار به یک ماشین تعلق بگیرد.
- قید دوم برای این است که مدت زمان انجام کار ها توسط یک ماشین ،از مدت زمان کل کار بیشتر نشود.
- قید اول و سوم و چهارم معادل شرط $x_{ij} \in \{۰, ۱\}$ هستند.

۴ ارایه یک برنامه ریزی جدید

مطابق نکته ی گفته شده در بخش ۲.۱ ، قید جدیدی به برنامه ریزی صحیح مان اضافه می کنیم.

$$\begin{array}{lrll} \text{Minimize} & t & & \\ \text{subject to} & \sum_{i \in M} x_{ij} & = 1 & \text{for all } j \in J \\ & \sum_{j \in J} d_{ij} x_{ij} & \leq t & \text{for all } i \in M \\ & x_{ij} & \geq 0 & \text{for all } i \in M, j \in J \\ & x_{ij} & = 0 & \text{for all } i \in M, j \in J \text{ with } d_{ij} > T \\ & x_{ij} & \in \mathbb{Z} & \text{for all } i \in M, j \in J. \end{array}$$

توضیح قید جدید: فرض کنید t_{opt} جواب بهینه ی برنامه ریزی صحیح اولیه باشد. در این صورت اگر $d_{ij} > t_{opt}$ باشد، در حالت بهینه کار j توسط ماشین i انجام نمی شود. یعنی در جواب بهینه $x_{ij} = ۰$ است. برنامه ریزی بالا پارامتر T را به عنوان ورودی می گیرد و در صورتی که $T \geq t_{opt}$ باشد، جواب بهینه ی برنامه ریزی اول و دوم برابرند.

- دقت کنید که مجموعه ی جواب های شدنی برنامه ریزی اول و دوم لزوما برابر نیستند. زیرا قید جدید ممکن است باعث شود برخی از جواب های شدنی برنامه ریزی اول، در برنامه ریزی دوم صدق نکنند. چون برنامه ریزی اول (مساله ی زمان بندی) در زمان چند جمله ای قابل حل نیست، برای یافتن جواب بهینه، برنامه ریزی دوم را آرام سازی کرده و حل می کنیم.
- دقت کنید که ما مقدار بهینه ی برنامه ریزی صحیح اول (t_{opt}) را نمیدانیم .(زیرا اگر میدانستیم نیاز به ارایه ی الگوریتم تقریبی نداشتیم). در ابتدا می توانید فرض کنید که $T = maxd_{ij}$ است. (با انتخاب این مقدار، قید اضافه بی اثر می شود و اگر برنامه ریزی اول جواب شدنی داشته باشد، برنامه ریزی آرام دوم هم جواب شدنی دارد). در انتهای کار، روشی برای انتخاب T مناسب ارایه میدهیم.

آرام سازی

با حذف کردن قید $x_{ij} \in \mathbb{Z}$، برنامه ریزی صحیح جدید را آرام سازی می کنیم. این برنامه ریزی خطی را $LPR(T)$ می نامیم.

$$\begin{array}{lrll} \text{Minimize} & t & & \\ \text{subject to} & \sum_{i \in M} x_{ij} & = 1 & \text{for all } j \in J \\ & \sum_{j \in J} d_{ij} x_{ij} & \leq t & \text{for all } i \in M \\ & x_{ij} & \geq 0 & \text{for all } i \in M, j \in J \\ & x_{ij} & = 0 & \text{for all } i \in M, j \in J \text{ with } d_{ij} > T. \end{array}$$

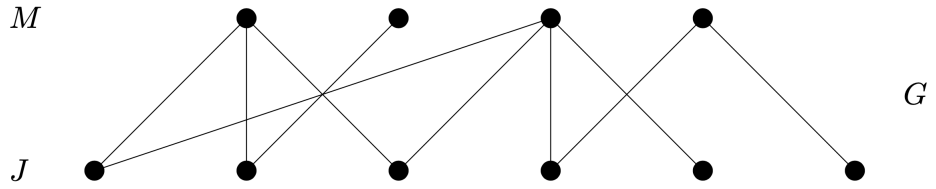
۵ ارایه ی یک الگوریتم تقریبی

فرض کنید برنامه ریزی خطی مان را با یک روش مناسب در زمان چند جمله ای حل کرده ایم(مثلا روش نقطه ی درونی) . و جواب بهینه ی حقیقی را به دست آورده ایم. میخواهیم از روی این جواب ، یک جواب صحیح که حداکثر دو برابر بزرگتر از جواب برنامه ریزی صحیح اولیه است بسازیم.

تعریف از روی جواب بهینه ی برنامه ریزی خطی، گراف دو بخشی $G = (V, E)$ را به شکل زیر تعریف می کنیم:

$$V = M \cup J$$

$$E = \{\{i, j\} \subseteq V | x_{ij} > 0\}$$



ماشین ها را در یک بخش و کار ها را در بخش دیگر قرار میدهیم . اگر برنامه ریزی خطی کار j را به ماشین i منتصب کرده بود $(x_{ij} > 0)$ ، یال $\{i, j\}$ را در گراف رسم می کنیم.

هدف چون برنامه ریزی خطی آرام سازی شده است، یک کار ممکن است بین چند ماشین تقسیم شده باشد. (هر راس از بخش کارها، به چند راس از بخش ماشین ها متصل شده باشد). ما میخواهیم از بین ماشین های متصل شده به کار j ، یکی را برای انجام این کار انتخاب کنیم به گونه ای که برای هر ماشین، فاصله ی زمانی آغاز تا پایان کار، کمتر از ۲ برابر جواب بهینه ی IP باشد.(این کار معادل ساختن یک جواب صحیح از روی جواب بهینه ی حقیقی است.)

لم ۱ در هر زیر گراف از گراف $G = (V, E)$ تعداد یال ها حداکثر برابر تعداد راس هاست.

$$|E'| \leq |V'| \qquad \text{for any subgraph of G}$$

اثبات در این اثبات فرض می کنیم برنامه ریزی LPR(T) دارای جواب شدنی است. ماتریس قیود LPR(T) را A بنامید و فرض کنید x^* یک جواب شدنی پایه (bfs) برای این برنامه ریزی باشد. با توجه به قیود برنامه ریزی خطی، ماتریس A یک سطر به ازای هر کار، یک سطر به ازای هر ماشین و یک سطر به ازای هر ماشین i و کار j ای که $d_{ij} > T$ است دارد. چون x^* یک bfs است ستون های A که متناظر با متغیر های $x^*_{ij} > 0$ هستند مستقل خطی اند.(این ستون ها در واقع متناظر با یال های G هستند) حال فرض کنید G' یک زیر ماتریس دلخواه از G با مجموعه ی راس های $M' \cup J' \subseteq M \cup J$ و مجموعه ی یال های $E' \subseteq E$ است. فرض کنید A' زیر ماتریسی از A با سطر های متناظر $M' \cup J'$ و ستون های متناظر E' است.

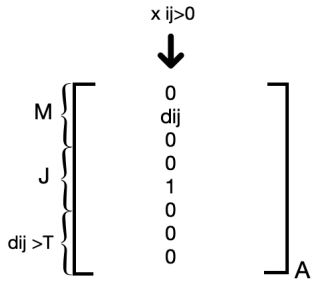
ادعا ستون های A' مستقل خطی اند.

اگر ادعا ثابت شود، رنک ستونی ماتریس برابر $|E'|$ میشود. چون رنک سطری برابر رنک ستونی است، A' باید حداقل به اندازه ی $|E'|$ سطر داشته باشد. درنتیجه $|M' \cup J'| = |V'| \geq |E'|$ و حکم ثابت می شود.

اثبات ادعا چون x^* یک bfs است ستون های متناظر با $x^*_{ij} > 0$ که $\{i,j\} \in E'$ است در A مستقل خطی اند. هر کدام از متغیر های $x^*_{ij} > 0$ که $\{i,j\} \in E'$ یک بار در قید مربوط به $i \in M'$ و یک بار در قید مربوط به $j \in J'$ ظاهر میشوند. به علاوه

$$x^*_{ij} > 0 \implies d_{ij} \leq T \implies x^*_{ij} \text{ در قید های مربوط به } d_{ij} > T \text{ ظاهر نمی شود.}$$

پس در ماتریس A ستون های مربوط E' در یکی از سطر های مربوط به M' و یکی از سطر های مربوط به J' ناصفر و در بقیه ی سطر ها صفر است.



چون این ستون ها در A مستقل خطی اند، با حذف سطر های صفر هم مستقل خطی می مانند. پس ستون های زیر ماتریس A' که سطر هایش متناظر با $M' \cup J'$ و ستون هایش متناظر با E' است، مستقل خطی اند.

لم ۲ اگر $T \geq 0$ به گونه ای باشد که $LPR(T)$ جواب شدنی داشته باشد و t^* جواب بهینه ی bfs برای $LPR(T)$ باشد، آنگاه میتوانیم یک زمانبندی با زمان حداکثر $t^* + T$ برای برنامه ریزی IP ارایه دهیم.

اثبات الگوریتم زیر یک زمان بندی با زمان مطلوب لم تولید می کند :

۶ مراحل الگوریتم تقریبی

گراف دو بخشی G متناظر با جواب بهینه ی برنامه ریزی خطی LPR(T) را در نظر بگیرید. میخواهیم در دو مرحله مشخص کنیم که هر کار توسط چه ماشینی انجام شود.

مرحله ی اول: کار های با درجه یک

در گراف G ،در بخش کار ها، راس هایی که درجه ی یک دارند را در نظر بگیرید.از آن جا که همه ی کار ها باید انجام شوند(طبق قید اول برنامه ریزی خطی) و کار های درجه ی یک فقط توسط یک ماشین انجام می شوند، این کار ها را به همان ماشین محول می کنیم.

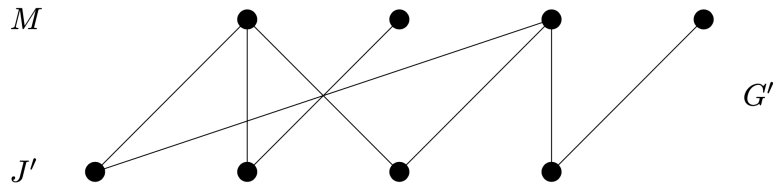
$$\Sigma_{j \in S_i} d_{ij} = \Sigma_{j \in S_i} d_{ij} x^*_{ij} \leq \Sigma_{j \in J} d_{ij} x^*_{ij} \leq t^*$$

- تساوی اول به این علت برقرار است که برای کار های با درجه یک $x^*_{ij} = 1$ است.
- نامساوی دوم از نامنفی بودن x^*_{ij} ها و d_{ij} ها نتیجه می شود.
- نامساوی سوم از قید دوم LPR(T) نتیجه می شود.

مرحله ی دوم : بقیه ی کار ها

ابتدا کار های با درجه ی یک و یال های متصل به آن ها را از گراف حذف می کنیم. این زیرگراف را G' مینامیم.

$$G' = (M \cup J', E')$$



نشان خواهیم داد که در G' تطابقی وجود دارد که همه ی کار ها را می پوشاند. اگر هر کار را بر اساس این تطابق به یک ماشین محول کنیم، آنگاه کل کار ها در زمان حداکثر T به پایان میرسند. زیرا در قید جدیدی که به برنامه ریزی اضافه کردیم، داشتیم:

$$x_{ij} = ۰ \qquad \text{for all } i \in M, j \in J \text{ with } d_{ij} > T$$

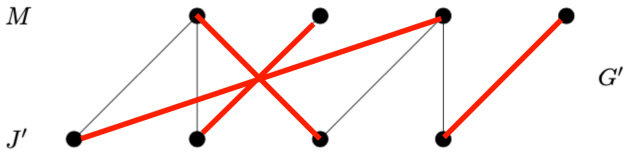
و طبق تعریف گراف برای هر یال $\{i,j\}$ داریم : $x_{ij} > ۰$. در نتیجه تنها کار هایی که در زمان کمتر از T انجام می شوند در گراف یال دارند. پس در تطابق پوشا،هر ماشین یک کار دارد و آن کار در زمان کمتر از T انجام می شود.

اثبات وجود تطابق پوشا

قضیه ی هال: در زیرگراف $G' = (M \cup J', E')$ یک تطابق پوشاننده ی J' داریم اگر و تنها اگر به ازای هر $J'' \subseteq J'$ داشته باشیم

$$|N(J'')| \geq |J''|$$

یعنی تعداد همسایه های J'' حداقل به اندازه ی J'' باشد.



قضیه ی هال را بدون اثبات می پذیریم و از آن برای اثبات وجود تطابق پوشا استفاده می کنیم. طبق قضیه ی هال کافیست نشان دهیم که برای هر $J'' \subseteq J'$ داریم $|N(J'')| \geq |J''|$. پس زیر مجموعه ی دلخواه $J'' \subseteq J'$ را در نظر بگیرید. چون زیرگراف G' از حذف کار های درجه ی یک به دست آمده است، درجه ی هر کار در این زیرگراف حداقل ۲ است. پس اگر e تعداد یال های خارج شونده از J'' باشد داریم

$$e \geq ۲|J''|$$

از طرفی طبق لم یک ،تعداد راس های هر زیرگراف G حداقل به اندازه ی تعداد یال هاست پس داریم

$$|J'' \cup N(J'')| \geq e \geq ۲|J''|$$

چون J'' و $N(J'')$ اشتراکی ندارند داریم

$$|J'' \cup N(J'')| = |J''| + |N(J'')| \geq ۲|J''| \implies |N(J'')| \geq |J''|$$

در نتیجه طبق قضیه ی هال یک تطابق پوشاننده ی J' برای زیرگراف G' وجود دارد.

زمان مرحله ی۲ + زمان مرحله ی ۱ = زمان اتمام کار ماشین

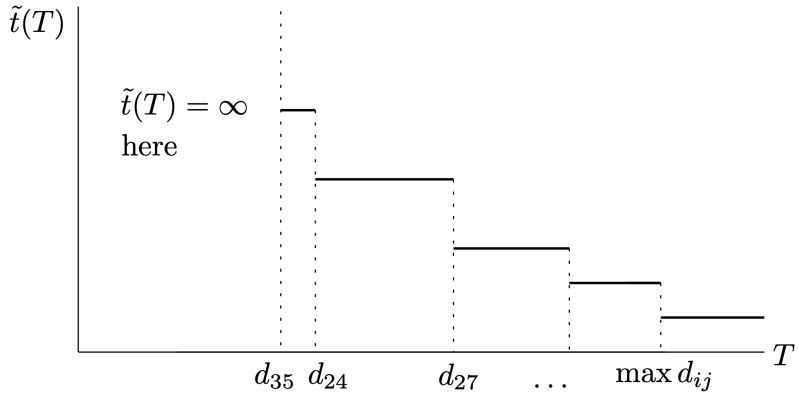
زمان مرحله ی ۱ $\leq t^*$

زمان مرحله ی۲ $\leq T$

بنابراین توانستیم از روی جواب بهینه ی برنامه ریزی خطی، یک زمان بندی با حداکثر زمان $t^* + T$ ارایه دهیم. قبلا خواندیم که مساله ی تطابق در زمان چند جمله ای قابل حل است. پس هر دو مرحله ی الگوریتم در زمان چند جمله ای به پایان میرسند.

۷ انتخاب T مناسب

گفتیم که برنامه ریزی آرام سازی شده T را به عنوان پارامتر ورودی میگیرد و احتمالا یک جواب بهینه ی وابسته به T تولید می کنید. نمودار جواب بهینه ی LPR(T) بر حسب T به شکل زیر است.



در این نمودار اگر T خیلی کوچک باشد، (مثلا $T < mind_{ij}$ باشد) برنامه ریزی LRP(T) جواب شدنی ندارد. با افزایش مقدار T مجموعه ی جواب شدنی این برنامه ریزی احتمالا بزرگتر میشود و در جایی مقدار $t^*(T) + T$ کمینه می شود. چون جواب الگوریتم تقریبی ما $t^*(T) + T$ بود، بهترین T ، آن است که $t^*(T) + T$ را کمینه کند

- نکته ی ۱ : این T قابل محاسبه است زیرا مطابق نمودار، مقدار $t^*(T)$ در $T = d_{ij}$ ها تغییر می کند. و از $T > maxd_{ij}$ ثابت می ماند. پس مینیمم $t^*(T) + T$ در یکی از $T = d_{ij}$ ها اتفاق می افتد. اگر مقدار تابع را در این نقاط محاسبه کنیم بین آن ها مینیمم بگیریم، T مناسب را می یابیم.
- نکته ی ۲ : دقت کنید که با انتخاب این T ،برنامه ریزی LPR(T) ممکن است یک آرام سازی برای برنامه ریزی صحیح اولیه مان نباشد. اما اهمیتی ندارد زیرا هدف ما ارایه ی یک جواب ۲ ـ تقریب برای مساله ی زمان بندی بود.

۸ اثبات ۲ ـ تقریب بودن

در بخش ۶ یک زمان بندی با زمان $t^*(T) + T$ ارایه دادیم و در بخش ۷ یک T پیدا کردیم که مقدار $t^*(T) + T$ را کمینه کند. اگر این T را T^* بنامیم میخواهیم ثابت کنیم $t^*(T^*) + T^* \leq ۲T_{opt}$
داریم:

$$t^*(T^*) + T^* = min_T(t^*(T) + T) \leq t^*(t_{opt}) + t_{opt}$$

نامساوی برقرار است زیرا $t^*(T^*) + T^*$ مقدار مینیمم تابع روی تمام T هاست و t_{opt} هم یکی از T هاست. از طرفی چون $LPR(t_{opt})$ یک آرام سازی برای برنامه ریزی صحیح است ، جواب بهینه ی آن کوچتر مساوی جواب بهینه ی برنامه ریزی صحیح است، یعنی $t^*(t_{opt}) \leq t_{opt}$. در نتیجه

$$t^*(T^*) + T^* = min_T(t^*(T) + T) \leq t*(t_{opt}) + t_{opt} \leq ۲t_{opt}$$

پس یک الگوریتم ۲ ـ تقریب ارایه کردیم که در زمان چند جمله ای یک جواب برای مساله ی زمان بندی پیشنهاد می کند. بهترین الگوریتم ارایه شده برای این سوال، یک الگوریتم $\frac{۳}{۲}$ ـ تقریب است. پس جواب الگوریتم ما به نظر جواب خوبی برای این مساله است.

۹ ارجاع و منابع

۱. Understanding and using Linear Programming , Matousek

۲. ویدیو جلسه ی شانزدهم