تمرین‌های الگوریتم تقریبی

سپیده آقاملائی

۹۲۲۰۱۹۴۱

۱. رنگ‌آمیزی رأسی: گراف بدون جهت $G = (V, E)$ داده شده است؛ رأسهای آن را با کمترین تعداد رنگ طوری رنگ کنید که دو سر هر یال رنگ‌های متفاوت داشته باشند.

(آ) یک الگوریتم حریصانه برای رنگ آمیزی گراف G با $\Delta + 1$ رنگ بدهید که در آن Δ بیشترین درجه رأسهای گراف G است.

با یک پیمایش عمقی گراف را رنگ می‌کنیم، به این صورت که راسی که در آن هستیم با رنگی متفاوت از همسایه‌های آن رنگ می‌کنیم. این کار امکان پذیر است، زیرا حداکثر درجه هر راس Δ است که با خود آن راس $\Delta + 1$ رأس می‌شود و ما به این تعداد رنگ داریم. حالتی که دو سر یک یال همرنگ باشد به وجود نمی‌آید، چون حداکثر یک رأس در حال رنگ شدن است و آن هم طوری رنگ می‌شود که رنگ آن با همسایه‌هایش متفاوت باشد.

(ب) یک الگوریتم بدهید که یک گراف ۳-رنگ‌پذیر را با $O(\sqrt{n})$ رنگ، رنگ‌آمیزی کند.

راهنمایی: به ازای هر رأس v زیرگراف القایی روی همسایه‌های آن ($N(v)$)، دوبخشی است و در نتیجه به صورت بهینه قابل رنگ‌آمیزی است. اگر v دارای درجه ی بیشتر از $\sqrt{n}$ بود آنگاه $v \cup N(v)$ را با ۳ رنگ متمایز رنگ کنید. این کار را ادامه بدهید تا همه ی راسهایی که درجه کمتر یا مساوی $\sqrt{n}$ دارند، رنگ‌آمیزی شوند. سپس از الگوریتم قسمت اول استفاده کنید.

طبق راهنمایی سوال عمل می‌کنیم. ثابت می‌کنیم می‌توانیم همه با استقرا روی رأسهایی که درجه‌ی آنها از $\sqrt{n}$ بیشتر است (به آنها در ادامه اثبات رأس درجه بالا می‌گوییم) مسأله را حل می‌کنیم.

حالت پایه وقتی است که هیچ رأس با درجه بیشتر از $\sqrt{n}$ نداشته باشیم که طبق قسمت الف حل می‌شود، چون تعداد رنگ‌ها $\sqrt{n}$ است که یکی بیشتر از ماکسیمم درجه رأسهای رنگ نشده است؛ چون درجه‌ی رأسهای رنگ نشده از $\sqrt{n}$ کمتر است.

فرض استقرا: می‌توانیم گراف با تعداد راس درجه بالای کمتر از k را با این الگوریتم رنگ کنیم.

حکم استقرا:می‌توانیم گراف با تعداد رأس درجه بالای k را با این الگوریتم رنگ کنیم.

یک رأس درجه بالا (مثل v را رنگ می‌کنیم. چون گراف ۳ رنگ‌پذیر است، زیرگراف القایی همسایه‌های آن دوبخشی خواهد بود به طور بهینه می‌توان آن را با دو رنگ، رنگ‌آمیزی کرد. (هر بخش را یک رنگ می‌کنیم.) اثبات: برهان خلف. چون گراف ۳ رنگ‌پذیر است، یک رنگ‌آمیزی از آن را در نظر می‌گیریم. رنگ رأس v را حذف کنیم، بقیه رأسها چون مجاور v هستند نمی‌توانند با آن همرنگ باشند، پس رنگ این رأسها با v متفاوت است. یعنی این رأسها فقط با دو رنگ، رنگ‌آمیزی شده‌اند یعنی زیرگراف همسایه‌های رأس v دوبخشی است. (تناقض)

با رنگ آمیزی این رأس یکی از تعداد رئوس درجه بالا کم می‌شود و طبق فرض استقرا می‌توان بقیه رئوس را رنگ کرد.

۲. پوشش سطلی: n شئ با اندازه‌های $a_1, ..., a_n \in (0, 1]$ داده شده است؛ تعداد سطل‌های ایجاد شده را ماکسیمم کنید طوری که جمع وزن اشیای هر سطل حداقل ۱ باشد. یک الگوریتم مجانبی PTAS برای این مساله بدهید که به ورودی‌هایی محدود شده باشد که اندازه اشیای آنها برای یک ثابت $c > 0$ حداقل c باشد.

راهنمایی: ایده‌ی اصلی الگوریتم ۹.۶ کتاب وزیرانی این مسأله را هم حل می‌کند.

ابتدا نسخه‌ی محدود شده‌ی مسأله را حل می‌کنیم.

فرض کنید تعداد وزن‌های متمایز اشیا K باشد. در این صورت حداکثر تعداد اشیای هر سطل $M = \lceil \frac{2}{c} \rceil$ است؛ چون جمع اشیای اضافی یک سطل از ۱ کمتر است (در غیر این صورت این اشیا را می‌توانستیم در یک سطل دیگر بگذاریم و تعداد سطل‌ها بیشتر شود که با بهینه بودن جواب متناقض است). پس تعداد وزن سطل‌های متمایز حداکثر$R = \binom{M+K}{K}$ است در نتیجه تعداد کل حالت‌های مسأله

$P = \binom{n+R}{R}$ است که از مرتبه‌ی $O(n^R)$ است که R فقط به c وابسته است که ثابت است یعنی می‌توانیم همه‌ی حالت‌های ممکن را چک کنیم و الگوریتم زمان چندجمله‌ای خواهد داشت.

حالا مسأله‌ی اصلی را حل می‌کنیم. برای تبدیل مسأله اصلی به مسأله‌ی قبل اشیا را مرتب می‌کنیم و آنها را به K دسته تقسیم می‌کنیم و اشیای هر دسته را به وزن مینیمم آن دسته رند می‌کنیم، در این صورت K وزن متمایز داریم و مسأله به حالتی که حل کردیم تبدیل می‌شود. برای تقسیم اشیا به K دسته آنها را به گروههای Q تایی تقسیم می‌کنیم (به ترتیب صعودی) که $Q = \lceil \frac{n}{K} \rceil$ است درنتیجه تعداد دسته‌ها K تا می‌شود. حداکثر تفاوت بین تعداد سطل‌های بهینه و تعداد سطل‌های الگوریتم Q است، چون این اختلاف حتما کمتر از اختلاف حالتی است که وزن اشیا به بالا رند شده باشد و وزن اشیا به پایین رند شده باشد. هر جواب حالت رند شده به پایین برای مسأله‌ی اصلی هم جواب است. هر جواب حالت رند شده به بالا را می‌توان با جایگزین کردن هر شئ با Q -امین شئ بزرگتر از آن به یک جواب برای مسأله اصلی بدون در نظر گرفتن Q شئ با کمترین وزن تبدیل کرد؛ چون وزن هر شئ در این حالت از ماکسیمم دسته‌ی آن که به آن رند شده است بیشتر می‌شود. تعداد سطل‌هایی که Q شئ با کمترین وزن پر می‌کردند از Q کمتر است (چون حداکثر وزن هر شئ ۱ است). پس حداکثر اختلاف حالت رند شده به پایین و بالا Q است پس حداکثر اختلاف جواب بهینه و جواب الگوریتم ما Q است چون تنها تفاوت در رند کردن است.

$$OPT \leq ALG + Q \leq OPT + Q \leq OPT + (c-1).OPT = c.OPT$$

پس باید Q را طوری تعیین کنیم که از $(c-1)OPT$ کمتر مساوی باشد و همچنین یک عدد صحیح باشد. می‌دانیم جواب بهینه از $\frac{n}{\frac{2}{c}}$ بیشتر/مساوی است، چون تعداد سطل‌ها از تعداد اشیا تقسیم بر کران بالای تعداد اشیای هر سطل بیشتر است. پس $Q = \lfloor n * c * (c-1)/2 \rfloor$ قرار می‌دهیم. از اینجا $K = \lfloor \frac{2}{c*(c-1)} \rfloor$ به دست می‌آید.

۳. برش-k بیشینه: یک گراف بدون جهت G=(V,E) با یالهای وزن‌دار نامنفی داده شده‌است، هر رأس را به صورت تصادفی به یکی از مجموعه‌های $S_1, ..., S_k$ اختصاص می‌دهیم، طوری که مجموع هزینه‌ی یالهایی که بین این مجموعه‌ها قرار می‌گیرند بیشینه شود. نشان دهید که متوسط تعداد یالهای بین این مجموعه‌ها حداقل $\frac{OPT}{2}$ است.

احتمال وجود هر یال برابر است با احتمال وجود دو سر آن در مجموعه‌های مجزا. هر رأس با احتمال $\frac{1}{k}$ در هر کدام از مجموعه‌ها قرار می‌گیرد، پس احتمال اینکه هر یال حضور داشته باشد $\frac{\binom{k}{2}}{k^2} = \frac{\frac{k*(k-1)}{2}}{k^2} = \frac{k-1}{2k}$ است. پس امید ریاضی تعداد یالهای بین این مجموعه‌ها برابر است با:

$$E(edges) = \Sigma_{(u,v)\in E} \frac{k-1}{2k} = \frac{k-1}{2k} * E \leq \frac{E}{2}$$

$$|E| \geq OPT \geq E(edges) = \frac{|E|}{2} = ALG$$

$$\Rightarrow OPT \geq \frac{|E|}{2} = ALG \geq \frac{OPT}{2}$$

۴. فروشنده دوره‌گرد متریک: نسخه‌ای از فروشنده دوره‌گرد متریک را در نظر بگیرید که در آن هدف پیدا کردن مسیر ساده‌ای شامل همه‌ی رأسهای گراف است. سه مسأله متفاوت بر اساس تعداد سرهای داده‌شده از مسیر به وجود می‌آیند: ۰، ۱ یا ۲. الگوریتم‌های تقریبی زیر را به دست آورید.

(آ) اگر ۰ یا ۱ سر مسیر مشخص شده باشد، یک الگوریتم با ضریب $\frac{3}{2}$ به دست آورید.

(ب) اگر هر دو سر داده شده باشد، یک الگوریتم با ضریب $\frac{5}{3}$ به دست آورید.

راهنمایی: از ایده‌ی الگوریتم ۳.۱۰ کتاب وزیرانی استفاده کنید.

ابتدا یک MST به دست می‌آوریم. سپس یک تطابق بین رأسهای درجه فرد و سرهای داده شده مسیر طوری پیدا می‌کنیم که درجه سرها فرد شود. یعنی اگر درجه فعلی سرهای مسیر زوج است در تطابق آنها را هم حساب می‌کنیم و در غیر این صورت آنها را در تطابق حساب نمی‌کنیم. اگر سرهای داده شده کمتر از ۲ بود به دلخواه سرهای دیگر را انتخاب می‌کنیم. سپس این یالها را به MST اضافه می‌کنیم و یک گراف نیمه اویلری به دست می‌آید که با عمل shortcutting یک مسیر همیلتونی در آن به دست می‌آوریم و آن را به عنوان جواب مسأله اعلام می‌کنیم.

می‌دانیم وزن MST از مسیر بهینه کمتر یا مساوی است، چون در غیر این صورت مسیر بهینه به عنوان MST پیدا میشد. در حالتی که ۰ یا ۱ سر داده شده باشد اجتماع یالهای تطابق کامل و دو تطابق برای رأسهای درجه فرد می‌دهد که بعد از طبقshortcutting نامساوی

مثلث وزن آن کمتر می‌شود. چون مسیر بهینه بعد از shortcutting دو تطابق کامل می‌دهد وزن مینیمم تطابق‌ها از نصف مسیر بهینه کمتر/مساوی است پس در این حالت جواب الگوریتم $OPT/2 + OPT = 3/2OPT$ به دست می‌آید. در حالتی که دو سر مسیر داده شده باشد اجتماع تطابق و درخت شامل سه تطابق کامل برای رأسهای درجه فرد است که بعد از عمل shortcutting کمتر از وزن دو تا از آنها باقی می‌ماند یعنی جواب الگوریتم $2/3OPT + OPT = 5/3OPT$ است.

۵. مسأله زمانبندی: n کار باید روی یک ماشین زمانبندی شوند که در آن هر کار j یک زمان پردازش p_j، یک وزن w_j و یک مهلت انجام d_j دارد؛ $j = 1, 2, ..., n$. هدف زمانبندی کارها به صورتی است که وزن کل کارهای انجام شده قبل از مهلت انجام را بیشینه کنیم. ابتدا ثابت کنید که همیشه یک زمانبندی بهینه وجود دارد که در آن همه‌ی کارهای on-time قبل از همه‌ی کارهای late انجام می‌شوند و کارهای on-time به ترتیب زودترین مهلت پایان انجام می‌شوند؛ از این نتیجه ساختاری استفاده کنید تا نشان دهید که چطور این مسأله را با برنامه‌نویسی پویا در زمان $O(nW)$ که در آن $W = \Sigma_j w_j$ است حل کنیم. سپس از این نتیجه استفاده کنید و یک FPTAS به دست آورید.

برای حل مسأله با برنامه‌نویسی پویا به این صورت عمل می‌کنیم که ابتدا کارها را بر اساس مهلت آنها مرتب می‌کنیم. سپس یک آرایه $W + 1$ عنصری در نظر می‌گیریم که در آن قرار است حداقل زمان لازم برای رسیدن به سود i در درایه i-ام آن ذخیره شود و مقدار خانه ۰ را ۰ و مقدار بقیه خانه‌ها را بینهایت قرار می‌دهیم. به ازای هر کار i مقادیر همه‌ی خانه‌های آرایه را با زمان کار فعلی جمع می‌کنیم و اگر از مهلت پایان آن کار کمتر یا مساوی بود، خانه متناظر مجموع سود خانه فعلی و سود این کار را با زمان فعلی مینیمم می‌گیریم. در پایان الگوریتم بزرگترین خانه آرایه که زمان بینهایت ندارد را به عنوان جواب مسأله برمی‌گردانیم. زمان این الگوریتم تعداد اشیا ضربدر تعداد خانه‌های آرایه است که $O(nW)$ است.

برای حالت FPTAS فرض می‌کنیم سود اندازه‌ی مسأله باشد. سود کارها را بر K تقسیم می‌کنیم و کف می‌گیریم. با این سودهای جدید مسأله را در حالت بهینه حل می‌کنیم. اختلاف سود واقعی و جدید هر کار کمتر از K است، چون:

$$w'_i = \lfloor \frac{w_i}{k} \rfloor \Rightarrow w_i - K * w'_i = K* \} \frac{w_i}{k} \{ < K$$

پس اختلاف جواب بهینه و جواب فعلی حداکثر nK است. می‌خواهیم اختلاف جواب بهینه و جواب الگوریتم ما از $\epsilon * OPT$ کمتر باشد و می‌دانیم که $OPT >$? پس اختلاف جواب بهینه و جواب الگوریتم کمتر از $W * \epsilon$ است. پس $K = \frac{W*\epsilon}{n}$. در این صورت زمان الگوریتم هم $O(nW')$ خواهد بود که 'w مجموع وزن اشیای رند شده است. یعنی داریم:

$$W' = \Sigma \lfloor \frac{w_i}{K} \rfloor \leq \frac{\Sigma w_i}{\Sigma K} = \frac{W}{nK} = \frac{1}{\epsilon}$$

پس زمان الگوریتم بر حسب $\frac{1}{\epsilon}$ و n خطی است. ضریب تقریب آن را هم خودمان با تنظیم مقدار K به مقدار مورد نظر رساندیم.

$$OPT - ALG < W * \epsilon$$

۶. مسأله نسبت جمع زیرمجموعه:

یک FPTAS برای این مسأله به دست بیاورید: n عدد صحیح مثبت $a_1 < a_2 < ... < a_n$ داده شده‌اند، دو زیرمجموعه ناتهی مجزای $S_1, S_2 \in \{1, ..., n\}$ که $\Sigma_{i \in S_1} a_i \geq \Sigma_{i \in S_2} a_i$ باشد پیدا کنید طوری که نسبت $\frac{\Sigma_{i \in S_1} a_i}{\Sigma_{i \in S_2} a_i}$ کمینه شود.

راهنمایی:ابتدا یک الگوریتم با زمان شبه چندجمله‌ای برای این مسأله به دست آورید. سپس به صورت مناسب مقیاس و رند کنید.

طبق نامساوی داده شده می‌فهمیم که همه‌ی جوابها از ۱ بیشتر/مساوی هستند. پس باید کاری کنیم که مجموع اعضای این مجموعه‌ها تا حد امکان به هم نزدیک باشد.برای این کار از برنامه‌نویسی پویا استفاده می‌کنیم. B را $\Sigma_{i=1} n a_i$ تعریف می‌کنیم. یک آرایه $t[0..n][0..B]$ که درایه‌ی سطر i و ستون b آن ۱ است در صورتی که زیرمجموعه‌ای از اشیای $a_1, ..., a_i$ وجود داشته باشد که جمع آن b شود و حتما شامل a_i باشد و در غیر این صورت ۰ است. این شرط شامل a_i بودن باعث می‌شود زیرمجموعه یکتا به دست بیاید. آرایه‌ی $c[0..n][0..B]$ شامل مجموعه‌ی یکتای متناظر آن در آرایه t است. الگوریتم را تا جایی ادامه می‌دهیم که دو عدد صحیح متمایز i_1, i_2 پیدا شوند که $t[i_1][j] = t[i_2][j] = 1$ باشد. که در الگوریتم ما کمترین j را به دست می‌آوریم. در غیر این صورت جدول‌ها کامل پر می‌شوند و ما نسبت آنها را مقایسه می‌کنیم تا کمترین نسبت را پیدا کنیم. زمان این الگوریتم $O(nB^2)$ است.

شمای تقریبی FPTAS برای مسأله: برای $m = 2, ..., n$ نمونه‌ای از مسأله I_m را نسخه‌ای از نسبت زیرمجموعه‌ها تعریف می‌کنیم که شامل m عدد کوچکتر باشد. به ازای هر ϵ در بازه‌ی $0 < \epsilon < 1$ داده شده، $K = \frac{\epsilon^2 . a_m}{2m}$ تعریف می‌کنیم. $n_0 \leq n$ را بزرگترین عدد صحیحی تعریف می‌کنیم که به ازای آن K از ۱ کمتر باشد. الگوریتم ما به این صورت است که به ازای $m \leq n_0$ الگوریتم قسمت قبل را اجرا می‌کنیم؛ چون $a_n 0 \leq \frac{2n}{\epsilon^2}$ است این کار زمان چندجمله‌ای می‌گیرد.

اگر $n_0 < m \leq n$ باشد، مسأله را ساده می‌کنیم تا تعداد اعداد متمایز آن چندجمله‌ای باشد. به ازای $i = 1, ..., m$ تعریف می‌کنیم $a'_i = \lfloor \frac{a_i}{K} \rfloor$. پس $a'_m = \lfloor \frac{2m}{\epsilon^2} \rfloor$ از مرتبه چندجمله‌ای است. نسخه‌ای از مسأله که شامل $a'_i \geq m/\epsilon$ است را I'_m می‌نامیم. اگر I'_m شامل t عدد $a'_{m-t+1}, ..., a'_m$ باشد. چون $\epsilon \leq 1$ است پس $a'_m \geq m/\epsilon$ است و در نتیجه $t > 0$ است. بر اساس مقدار t مسأله را به دو حالت تقسیم می‌کنیم. اگر $t = 1$ باشد آنگاه جواب به صورت $m\{$ و $\}j, j+1, ..., m-1\{$ است که j کوچکترین عدد صحیحی است که $a_{j+1} + ... + a_{m-1} < a_m$ است. اگر $t > 1$ باشد آن را به طور دقیق با الگوریتم شبه چندجمله‌ای داده شده (برنامه‌نویسی پویا) حل می‌کنیم. اگر جواب بهینه‌ی آن ۱ بود الگوریتم آن را برمی‌گرداند؛ اگر بزرگتر از ۱ بود الگوریتم مجموعه زیرمجموعه‌هایی می‌سازد که مجزا هستند و مجموع آنها متمایز است.

$$a'_m \leq 2m/\epsilon^2 \Rightarrow \Sigma i = m - t + 1 m a'_i < 2m^2/\epsilon^2 \Rightarrow 2^t \leq 2m^2/\epsilon^2 \Rightarrow t \leq 2log(m/\epsilon) + 1$$

در حالت اول داریم:

$$\Sigma i \in S_1 a_i / \Sigma i \in S_2 a_i \leq 1 + a_j/a_m < 1 + \epsilon$$

در حالت دوم داریم:

$$\frac{\Sigma i \in S_1 a_i}{\Sigma i \in S_2 a_i} leq \frac{\Sigma i \in S_1 K.(1 + a'_i)}{\Sigma i \in S_2 K.a'_i} = 1 + \frac{|S_1|}{\Sigma i \in S_2} \leq 1 + \frac{t}{m/\epsilon} < 1 + \epsilon$$

$$\Sigma i \in S_1 a_i / \Sigma i \in S_2 a_i \leq 1 + a_j / \Sigma i \in S_2 a_i \leq 1 + a_j/a_m < 1 + \epsilon$$

۷. گراف بدون دور: فرض کنید یک گراف جهت‌دار بدون دور با گره منبع s و گره مقصد t داده شده است و هر یال جهت‌دار e هزینه c_e و طول L_e دارد. همچنین کران L روی طول یالها داده شده است. یک FPTAS برای مسأله‌ی پیدا کردن کران L بدهید. یک PTAS برای مسأله پیدا کردن مسیر با کمترین هزینه از s به t به طول حداکثر 'L بدهید.

ابتدا قسمت اول مسأله را حل می‌کنیم و بعد با روش هرس پارامتری قسمت دوم آن را حل می‌کنیم.