به نام خدا

ساختمان داده‌ها و الگوریتم‌ها (۴۰۲۵۴)
دانشگاه صنعتی شریف
مدرس: دکتر مهدی صفرنژاد

آزمون پایان‌ترم
انتشار: ۲۶ دی ۱۴۰۰

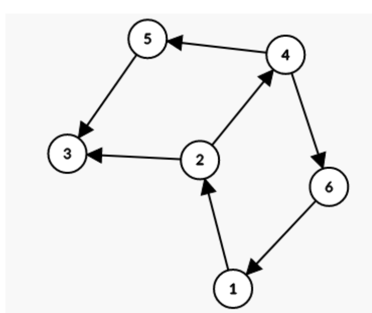
آزمون پایان‌ترم

زمان این امتحان ۱۲۰ دقیقه است. امتحان از ۱۰۰ نمره است. توضیحات ابتدای سؤالات را به دقت بخوانید و پاسخ بخش‌های مختلف سؤال را بنویسید. ۵ دقیقه زمان آپلود در نظر گرفته‌شده‌است.
امیدواریم تا این‌جای کلاس با تفکر الگوریتمی آشنا شده باشید و با همین تفکر به سؤالات امتحان به خوبی پاسخ دهید.

سؤال ۱. [۲۲ نمره]

آ. [۱۵ نمره] در گراف جهت‌دار $G = (V, E)$، هر یک از رئوس دارای وزن $w(v) \in \mathbb{Z}$ هستند. همچنین در G هیچ دوری اعم از جهت‌دار و غیرجهت‌دار وجود ندارد وجود ندارد. کران می‌خواهد یکی از رئوس این گراف را انتخاب کند. به ازای هر رأس مثل v که کران انتخاب می‌کند، به او به اندازه‌ی مجموع وزن تمام رئوس در G که به v مسیر جهت‌دار دارند شکلات می‌دهند (توجه کنید که ممکن است وزن رئوس منفی باشد و از کران شکلات بگیرند). با استفاده از الگوریتمی با پیچیدگی زمانی $O(|V| + |E|)$ به کران کمک کنید تا رأسی را انتخاب کند که بیشترین شکلات ممکن را به دست بیاورد (و یا در صورتی که این کار ممکن نیست، کمترین تعداد شکلات را از او بگیرند).

ب. [۷ نمره] مؤلفه‌های قویاً همبند گراف زیر را به کمک الگوریتم Kosaraju به دست بیاورید (روش استفاده‌شده را به اختصار شرح دهید و نتیجه‌ی هر قسمت از الگوریتم را بگویید).



پاسخ: آ. رئوس را به طور توپولوژیکال مرتب می‌کنیم. پس اگر v_i به v_j یال داشته باشد، حتما $i < j$. مجموع وزن برای یک رأس، برابر مجموع وزن رئوسی که به آن یال دارند و مجموع وزن برای آن رئوس است، در این شمارش، رأس‌هایی

که به چند تا از چنین رأس‌هایی یال داشته باشند، چند بار شمرده می‌شوند، اما باید توجه کنیم که اگر راسی به دو طریق شمرده شود، یک مسیر بدون جهت در گراف تشکیل می‌شود که مطابق فرض سوال در گراف وجود ندارد.

با توجه به این موضوع، می‌توانیم مقدار مجموع را برای هر راس به ترتیب توپولوژیکال پیدا کنیم، و از این مجموعه بیشینه را انتخاب کنیم.

ب. در ابتدا رئوس را به طور توپولوژیکال با شروع از رأس ۲ (به طور متناظر هر رأس دیگری) مرتب می‌کنیم. در این روال اگر ابتدا رئوس با مقادیر کمتر را ببینیم، راس‌ها را به ترتیب $(2, 3, 2, 4, 5, 4, 6, 1, 6, 4, 2)$ می‌بینیم و رئوس به ترتیب $(2, 4, 6, 1, 5, 3)$ در لیست قرار می‌گیرند (هر رأس در آخرین باری که دیده می‌شود به ابتدای لیست اضافه می‌شود).

در ادامه از ابتدای لیست بر اساس یال‌های ورودی رئوس حرکت می‌کنیم تا مولفه‌های قویا همبندی آن‌ها مشخص شود. در نتیجه مولفه‌ها به صورت $\{(2, 1, 6, 4), (5), (3)\}$ خواهند بود.

سؤال ۲. [۱۰ نمره] گراف وزن‌دار و جهت‌دار $G = (V, E)$ و راس $s \in V$ را در نظر بگیرید. می‌دانیم تنها یال‌هایی که مبدأ آن‌ها s است می‌توانند وزن منفی داشته باشند و وزن بقیه‌ی یال‌ها مثبت است. همچنین گراف G دور جهت‌دار منفی ندارد. آیا الگوریتم dijkstra برای پیدا کردن کوتاه‌ترین مسیر در گراف G از مبدأ s به درستی عمل می‌کند؟ ادعای خود را ثابت کنید.

پاسخ: گراف $G' = (V, E')$ را از روی G می‌سازیم با این تفاوت که به هر یک از یال‌های خروجی s به میزان $x = \min_{v \in N^+(v)} w(\{s, v\})$ اضافه می‌کنیم. واضح است که هر مسیری با مبدأ s در G' طولش دقیقا x واحد از مسیر متناظرش در G بیشتر است.

الگوریتم dijkstra را بر روی این دو گراف دنبال می‌کنیم و نشان می‌دهیم با هم متناظرند. به استقرا نشان می‌دهیم بعد از اولین مرحله

$$(v, w) \in Q_G \iff (v, w + x) \in Q_{G'}$$

و

$$v \neq s \implies w_G(v) = w_{G'}(v), w_G(s) = 0, w_{G'}(s) = x$$

که Q_G صف مربوط به الگوریتم بر روی گراف G و $w_G(v)$ بهترین وزن پیداشده برای v در گراف G است.

در مرحله‌ی اول در هر دو صف تنها مقدار $(s, 0)$ را دارند. پس از اولین گام، تمام همسایه‌های s با وزنی معادل وزن یال‌های متناظرشان به صف اضافه می‌شوند. پس $Q_G = \{(v, w) | e = (s, v) \in E, w = w_E(e)\}$ و $Q_{G'} = \{(v, w) | e = (s, v) \in E', w = w_{E'}(e)\}$. وضعیت $w_G(v)$ واضح است.

چون دور منفی نداریم s هیچ‌گاه وزنی بهتر از صفر نخواهد داشت و به صف اضافه نمی‌شود. پس راس v که در ابتدای صف بود، s نیست. همچنین توضیح دادیم s به هر حال به صف اضافه نمی‌شود. در موارد دیگر $w_G(v) < w + w_E(\{v, u\}) \iff w_{G'}(v) = w_G(v) + x < w + x + w_{E'}(\{v, u\}) = w + x + w_E(\{v, u\})$ پس گسترش رئوس به طور مشابه اتفاق می‌افتد.

پس در نهایت مسیرهای یکسانی با اختلاف وزن x در دو الگوریتم یافت می‌شود. پس در G همان مسیر و وزنی یافت شد که توضیح دادیم کوتاه‌ترین مسیر است.

سؤال ۳. [۲۲ نمره] فرض کنید عماد تابع درهم‌ساز $h(x) = x \mod 9$ و یک جدول درهم‌سازی با ۹ خانه را در اختیار دارد. در صورتی که عماد از زنجیر برای رفع برخورد استفاده کند،

آ. [۵ نمره] جدول عماد پس از اضافه کردن عناصر $\langle 33, 54, 69, 74, 18, 19, 47 \rangle$ به ترتیب را رسم کنید.

ب. [۵ نمره] آتنا همان جدول و تابع درهم‌ساز عماد را دارد، اما از روش آدرس‌دهی باز[۱] برای رفع برخورد استفاده می‌کند. جدول آتنا را پس از اضافه کردن همان اعداد قسمت قبل رسم کنید.

پ. [۷ نمره] آتنا تصمیم گرفته عدد 19 را از جدولش پاک کند. به او توضیح دهید که چطور می‌تواند این کار را انجام دهد و بعد از این کار چطور عمل جست‌وجو و اضافه کردن را انجام دهد.

ت. [۷ نمره] علی می‌خواهد تعدادی از جایگشت‌های nتایی به ازای یک $n < 10$ را به کمک تابع و جدول درهم‌سازی عماد ذخیره کند. او برای نشان دادن هر جایگشت، ارقام آن را به ترتیب به هم می‌چسباند تا یک عدد تشکیل دهد (مثلا عدد متناظر با جایگشت $\langle 2, 4, 1, 3 \rangle$ برابر 2413 خواهد بود). توضیح دهید که آیا جدول و تابع عماد، انتخاب خوبی برای کاربرد علی هستند و یا خیر (فرض کنید که علی می‌خواهد حتماً جایگشت‌هایش را در یک جدول با ۹ خانه نگه‌داری کند).

پاسخ: آ.

۰	$\langle 54, 18 \rangle$
۱	$\langle 19 \rangle$
۲	$\langle 74, 47 \rangle$
۳	$\langle \rangle$
۴	$\langle \rangle$
۵	$\langle \rangle$
۶	$\langle 33, 69 \rangle$
۷	$\langle \rangle$
۸	$\langle \rangle$

ب.

[۱] open-addressing

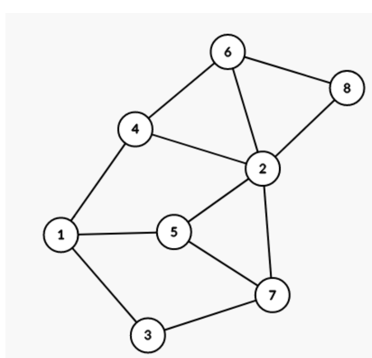
۰	54
۱	18
۲	74
۳	19
۴	47
۵	
۶	33
۷	69
۸	

پ. برای حذف، مقدار خانه‌ای که 19 در آن قرار گرفته (۳) را با یک مقدار مشخص مثل $NULL$ پر می‌کنیم، و در هنگام جستجو با دیدن آن متوقف نمی‌شویم، اما در هنگام اضافه کردن، در صورتی که به این مقدار رسیدیم، مقدار را در همان خانه قرار می‌دهیم.

ت. چون باقی‌مانده بر ۹ تمام جایگشت‌ها برابر باقی‌مانده بر ۹ مجموع ارقامشان است، تمام جایگشت‌ها در یک خانه قرار می‌گیرند و برای هر جستجو نیاز به جستجو در تمام جایگشت‌ها هستیم.

سؤال ۴. [۲۵ نمره]

آ. [۵ نمره] صبا یک گراف G مطابق شکل زیر دارد. او در این درخت عدد 6 را گم کرده است. صبا به جست‌وجوی عمق‌اول[۲] علاقه‌ی زیادی دارد، به همین دلیل می‌خواهد به کمک این جست‌وجو رأس 6 را در درختش پیدا کند. او جست‌وجویش را از رأس 3 شروع می‌کند و بین رأس‌هایی که نیاز است بین آن‌ها انتخاب کند، به رأس با مقدار کوچک‌تر می‌رود. به صبا بگویید به ترتیب چه رأس‌هایی را طی کند تا به رأس 6 برسد.



[۲] DFS

ب. [۱۰ نمره] کیمیا با دیدن جست‌وجوی نیمه‌کاره‌ی صبا، تصمیم گرفت که آن را تا آخرین رأس ادامه دهد. با این کار متوجه شد که تمام یال‌هایی که در درخت DFS طی نشده‌اند، یک رأس را به یکی از اجدادش متصل می‌کند. کیمیا می‌خواهد بداند که آیا می‌توان گفت که در هر گراف $G = (V, E)$ و T که یک درخت DFS از گراف G است، به ازای هر یال $e = (u, v) \in E$ یا u در T یکی از اجداد (و یا پدر) v است و یا برعکس. به او در پیدا کردن پاسخ این پرسش کمک کنید.

پ. [۱۰ نمره] هاشم برای انجام تمرینش تصمیم گرفته است که یک پیمایش DFS برای گراف بدون جهت $G = (V, E)$ پیدا کند. عماد برای کمک به او، یک جایگشت از اعضای V به او داده و ادعا کرده که این یک پیمایش DFS از گراف G است. هاشم به عماد اعتماد ندارد، پس از شما خواسته تا الگوریتمی با پیچیدگی زمانی $O(|V| + |E|)$ ارائه دهید که مشخص کند آیا جایگشت عماد یک پیمایش معتبر DFS از گراف G هست یا خیر.

پاسخ: آ. $6 \to 7 \to 5 \to 2 \to 4 \to 1 \to 3$

ب. یال (u, v) را در نظر بگیرید. بدون از دست دادن کلیت مسئله فرض کنید ابتدا v در پیمایش DFS دیده شده است.

پس در زمانی که DFS بر روی راس v شروع می‌شود، راس u دیده نشده است. چون بین u و v یال قرار دارد، حتما قبل از به پایان رسیدن DFS برای راس v راس u دیده می‌شود پس u از نوادگان v خواهد بود.

پ. در هر مرحله به کمک ورودی سعی می‌کنیم پیمایش DFS را ادامه دهیم. پس با شروع از راس اولیه، در صورتی که راس کنونی، به راس بعدی در دنباله یال داشته باشد به راس بعدی می‌رویم. همچنین مانند الگوریتم DFS راس‌هایی که دیده‌ایم را مارک می‌کنیم. علاوه بر این در هر مرحله، تا زمانی که تمام فرزندان یک راس دیده شده‌اند به راس پدرش برمی‌گردیم.

برای چک کردن وجود یک راس می‌توان یال‌ها را در HashMap از ماتریس مجاورت نگه داشت. و برای چک کردن به پایان رسیدن همسایه‌های یک الگوریتم، بررسی می‌کنیم که تمام همسایه‌های آن مارک شده باشند. (دقت کنید که این کار را حداکثر یک بار به ازای هر راس انجام می‌دهیم).

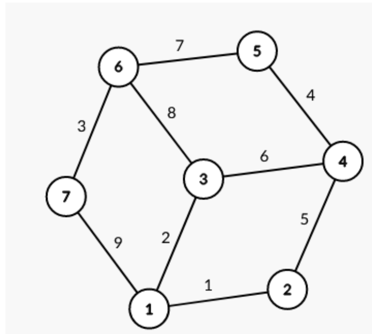
پیچیدگی زمانی این الگوریتم مشابه الگوریتم DFS از $O(|V| + |E|)$ خواهد بود.

سؤال ۵. [۵ نمره] سجاد گراف زیر را برای شما آورده، تا به ترتیب شماره‌ی یال‌ها، روی دو سر یال عملیات اجتماع را انجام دهید، و در هنگامی که مجموعه‌ی مربوط به یک راس ساخته نشده بود، عملیات ساخت مجموعه را روی آن راس نمایش دهید. این عملیات را به کمک نگه‌داری اعضای مجموعه‌ها در لیست پیوندی انجام دهید.

پاسخ:

$[1], [2]$

$[1 \to 2]$



$[1 \to 2][3]$

$[1 \to 2 \to 3]$

$[1 \to 2 \to 3][6][7]$

$[1 \to 2 \to 3][6 \to 7]$

$[1 \to 2 \to 3][6 \to 7][4][5]$

$[1 \to 2 \to 3][6 \to 7][4 \to 5]$

$[1 \to 2 \to 3 \to 4 \to 5][6 \to 7]$

4 و 3 در مجموعه‌ی یک هستند.

$[1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7]$

6 و 3 در یک مجموعه‌اند.

1 و 7 در یک مجموعه‌اند.

سؤال ۶. [۱۰ نمره] کیمیا به همراه خود m کوئری دارد که هر کدام از آن‌ها یک مقدار $a_i \in \mathbb{N}$ دارند. او از سجاد می‌خواهد که یک آرایه‌ی n تایی تهیه کند و تا جایی که می‌تواند، کوئری‌های کیمیا را در آرایه‌اش جا دهد. کیمیا دو شرط هم برای جا دادن کوئری‌ها در آرایه دارد. اول این که در هر خانه‌ی آرایه‌ی سجاد، حداکثر یک کوئری قرار بگیرد، و دوم آن که اگر کوئری i در خانه‌ی jام آرایه‌ی سجاد قرار گرفته، حتما $a_i < j$ باشد. به سجاد کمک کنید تا با الگوریتمی با پیچیدگی زمانی $O(n + m\log n)$ بیشترین تعداد کوئری را در آرایه‌اش جا دهد.

پاسخ:
واضح است که بهتر است هر کوئری را در بیشینه خانه‌ی ممکن قرار داد، و تا زمانی که این ویژگی را داریم، فرقی نمی‌کند اگر یک کوئری را با کوئری دیگری که قبلا در آن‌جا بوده جابه‌جا کنیم. پس هر کوئری را در بیشینه جای ممکن در صورت امکان قرار می‌دهیم.

بازه‌های متوالی از کوئریهای انجام شده را به صورت یک مجموعه می‌بینیم. همچنین، در نماینده‌ی مجموعه، کمینه‌ی مقدار موجود در مجموعه را نگه می‌داریم.
برای هر کوئری جدید در صورتی که مجموعه‌ای برای a_i ساخته نشده باشد، این مجموعه را می‌سازیم. در صورتی که چنین مجموعه‌ی وجود داشته باشد، و کمینه‌ی آن m باشد، در صورت وجود، مجموعه‌ی $m-1$ را می‌سازیم و این کوئری را در خانه‌ی $m-1$ قرار می‌دهیم.
در ادامه مجموعه‌ی $m-1$ را در صورت وجود با مجموعه‌های $m-2$ و m اجتماع می‌گیریم تا تمام مجموعه‌ها بازه‌های متوالی پری را نشان دهند که در دو سرشان کوئری قرار نگرفته.
از n ساخت مجموعه و m اجتماع استفاده شده است. پس پیچیدگی زمانی از $O(n + m\log n)$ خواهد بود.
علاوه بر این می‌توان از انتهای آرایه به کمک یک صف شروع کرد، و کوئری‌هایی که مقدار آن‌ها برابر خانه‌ی کنونی‌ست در صف قرار داد، سپس در صورت خالی نبودن صف، مقدار ابتدایی آن را در خانه‌ی آرایه قرار داد. پیچیدگی زمانی این الگوریتم از $O(n+m)$ خواهد بود.

موفق باشید