به نام یکتای هستی بخش

دانشگاه تهران، دانشکده‌ی مهندسی برق و کامپیوتر

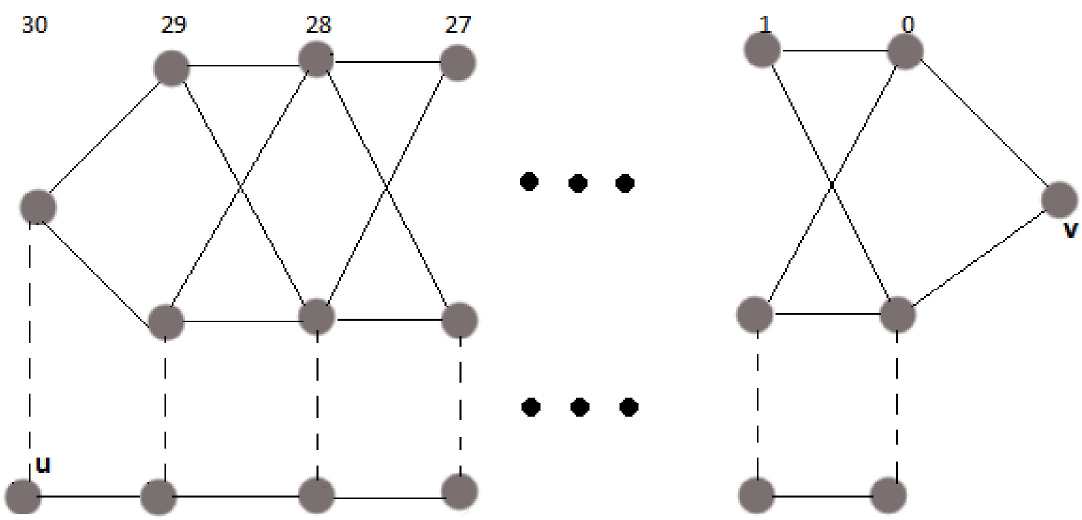
ساختمان داده‌ها و الگوریتم‌ها، نیمسال اول، سال تحصیلی ۹۴-۹۳

پاسخ تمرین شماره ۶

---

۱. دو حالت وجود دارد: ۱- تمام رئوس درجه‌ی زوج داشته باشند. در این صورت پاسخ همان تور اویلری است. زیرا هر یال را دقیقا یک بار ملاقات می‌کنیم. O(V+E) ۲- دو راس درجه فرد با نام‌های u, v داشته باشیم. در این حالت درجه‌ی تمام رئوس را به زوج تبدیل می کنیم تا امکان ملاقات تمام رئوس و بازگشت به راس آغازین ایجاد شود. برای این کار باید یک مسیر از u به v را به گراف اولیه اضافه کنیم. برای رسیدن به کم‌وزن‌ترین گشت، باید وزن این مسیر کمینه باشد. بنابراین کافیست به کمک الگوریتم dijkstra کوتاهترین مسیر بین u,v را بیابیم. O(E+V*logV)

۲. روش ساخت گراف به صورت زیر است:



وزن تمام یال‌ها برابر یک و تعداد کل رئوس استفاده شده برابر ۹۳ است. همچنین تعداد کوتاهترین مسیرها از راسی در لایه‌ی j برابر 2^j است. حال عدد k را در مبنای دو می‌بریم. داریم $۲^{۳۰} > ۱۰^{۹}$ بنابراین عدد k در مبنای دو حداکثر ۳۰ رقمی خواهد بود. سپس اگر بیت j ام آن یک بود، خطی که در لایه‌ی j ام با نقطه‌چین منایش داده شده است را وصل می کنیم.

۳. Bottleneck بین دو راس u, v را با bt(u, v) نمایش می‌دهیم. ادعا می‌کنیم به ازای هر u, v در گراف، bt(u, v) در MST آمده است. با برهان خلف ثابت می‌کنیم. MST را در نظر بگیرید و فرض کنید bottleneck تنها مسیری که داخل MST بین دو راس u, v وجود دارد، یال e باشد. طبق فرض برهان خلف مسیر دیگری به نام P بین u, v وجود خواهد داشت که bottleneck کمتر دارد. به این ترتیب تمام یال‌های موجود در مسیر P، وزنی کمتر از e خواهند داشت. حال یال e را از MST حذف کرده و گراف به دو مولفه‌ی همبندی تقسیم می‌شود. از راس u شروع کرده و روی مسیر P حرکت می‌کنیم. در طول مسیر قطعا یالی به نام e’ وجود خواهد داشت که بین دو مئولفه‌ی همبندی قرار بگیرد. اگر یال e’ را به MST-e اضافه کنیم، به یک درخت پوشای دیگر می‌رسیم که وزنی کمتر از MST اولیه دارد که با کمینه بودن MST اولیه در تناقض است. بنابراین حکم ثابت شد.

طبق حکم بالا، برای بدست آوردن bt(u, v) به ازای تمام رئوس، ابتدا MST را بدست می‌آوریم و سپس به ازای هر راس یک بار پیمایش BFS از آن راس را انجام داده و ماکزیمم یال را بین ریشه و بقیه رئوس بدست می‌آوریم. چون تعداد یال‌ها در MST برابر V-1 است، مرتبه‌ی اجرای هر BFS برابر O(V) خواهد بود.
مرتبه‌ی الگوریتم: O(E + V*logV + V*V)

۴. الف) ابتدا وزن تمام یال ها را قرینه کرده و سپس MST را در گراف حاصل پیدا می‌کنیم. یال‌هایی که در MST روی گراف جدید بدست می‌آیند، همان یال‌هایی هستند که در درخت پوشای بیشینه‌ی گراف اولیه خواهند بود.

ب) طبق تعریف صورت سوال G-F یک جنگل خواهدبود. برای کمینه کردن F باید G-F را بیشینه کنیم. در صورتی که هیچ یال منفی در درخت پوشای بیشینه وجود نداشته باشد، G-F بیشینه همان درخت پوشای بیشینه خواهد بود. در غیر این صورت G-F بیشینه برابر است با درخت پوشای بیشینه بجز یال های منفی آن. زیرا با حذف آنها G-F همچنان جنگل می‌ماند و وزنش بزرگتر خواهد شد.

۵. ابتدا با الگوریتم Johnson تمام یال ها را مثبت می‌کنیم. سپس از هر راس مانند v، dijkstra زده و درخت حاصل از اجرای الگوریتم را نگه می‌داریم. هر راس به تعداد رئوس موجود در زیردرختش، در کوتاهترین مسیر های بدست آمده از راس v به بقیه‌ی رئوس، ظاهر شده است این مقادیر را در آرایه‌ی P نگهداری می‌کنیم. برای یافتن تعداد رئوس در زیردرخت هر راس کافیست یکبار روی درخت حاصل DFS بزنیم. تعداد رئوس موجود در زیردرخت هر راس برابر است با مجموع تعداد رئوس موجود در هر یک از فرزندانش بعلاوه‌ی یک.(خودش را نیز در زیردرختش در نظر می‌گیریم.) چون DFS را روی یک درخت V راسی صدا می‌زنیم، از O(V) خواهد بود.
به ازای تمام رئوس این مراحل را انجام می‌دهیم. در نهایت راسی که بیشترین مقدار را در آرایه‌ی P داشته باشد، جواب مساله است. $O(V*(E + V*logV + V)) = O(V*E + V^2*logV)$

۶. الف) $E = O(V)\ , Dijkstra = O(E + VlogV) \Rightarrow order: O(V + VlogV) = O(VlogV)$
ب) با O(V) قابل حل است. ابتدا با یک پیمایش BFS کوتاهترین مسیر از ریشه به تمام رئوس را بدست می‌آوریم. (دقیقا یک مسیر از ریشه به هر راس وجود دارد) سپس با یک پیمایش DFS کوتاهترین مسیر از هر راس به ریشه را نیز پیدا می کنیم. در مورد برگ‌ها، پاسخ همان وزن یال برگ به ریشه است و در مورد رئوس غیر برگ(s) برابر است با:

$$\forall\, t|(s,t) \in E: min\ \{W(s,t) + dist(t)\}$$

در نهایت یک پیمایش BFS از u به v انجام می‌دهیم. اگر در مسیر پیدا شده از u به v، ریشه وجود نداشت، جواب طول همین مسیر است. در غیر این صورت برای رسیدن به راس v با شروع از u حتما باید از ریشه عبور کنیم. در این حالت جواب برابر است با dist(u) + طول مسیر از ریشه به v

۷. در این مساله به دنبال بزرگترین زیرمجموعه از رئوس هستیم که بین هر دو راس آن مسیر وجود داشته باشد. (i->j یا j->i). اگر گراف DAG باشد، پاسخ طولانی‌ترین مسیر موجود در آن است. در غیر این صورت ابتدا با دو DFS مولفه‌های قویا همبند گراف را پیدا می‌کنیم و به ازای هر مولفه‌ی قویا همبند یک راس با برچسبی که برابر تعداد

راس‌های آن مولفه است قرار می‌دهیم. سپس topological sort می‌زنیم و از راس آخر شروع کرده (راسی که یال خروجی ندارد) و یکی یکی به عقب می‌رویم. بزرگترین مسیر خارج شده از هر راس برابر است با ماکزیمم بزرگترین مسیرهای خارج شده از رئوسی که به آن‌ها یال دارد به علاوه‌ی برچسب خودش. به این ترتیب با O(V+E) بزرگترین خوشه بدست می‌آید.