دانشکده‌ی علوم ریاضی

آنالیز الگوریتم ها مدرس: دکتر شهرام خزایی

تمرین سری سوم

نام و نام‌خانوادگی : کژال باغستانی *شماره دانشجویی:* ۴۰۱۱۰۰۰۷۱

پرسش ۱

برای این سوال از تکنیک حریصانه استفاده میکنم به این صورت که در ابتدا از ریشه درخت شروع کرده و مرحله مرحله پایین می آیم تا به برگ ها برسم و به هر چراغ خاموشی که رسیدم آن را روشن میکنم.

مراحل:

۱. ابتدا از ریشه درخت BFS را اجرا کرده و درخت BFS گراف داده شده را تشکیل میدهم و راس های هر طبقه را در یک لیست مربوط به آن طبقه نگه میدارم برای مثال لیست ۴ مربوط به راس هایی به فاصله چهار از ریشه درخت است.

۲. از لیست شماره ۰ به ترتیب شروع کرده و هر راس که چراغ آن خاموش بود روشن میکنم تا به برگ ها برسم.

۳. تعداد چراغ هایی که روشن کردم را به عنوان پاسخ مساله خروجی میدهم.

اثبات درستی:
ادعا۱ :
به ازای هر چراغ خاموش برای روشن شدن حتما باید یا خود چراغ یا یکی از اجداد آن در درخت BFS کلیک شود.
اثبات:
بدیهی.

اتمام اثبات :

فرض کنیم که الگوریتم برای درخت به ارتفاع $n-1$ مینیمم تعداد سوییچ ها را درست خروجی میدهد. نشان میدهم برای درخت به ارتفاع n نیز این عدد را درست نشان میدهد. فرض کنیم درخت به ارتفاع n داریم که r فرزند دارد. اگر ریشه این درخت روشن باشد که تعداد مینیمم برای این درخت طبق الگوریتم $\sum_{i=1}^{r} min(Alg(T_i))$ مجموع تعداد مینیمم های زیر درخت های فرزندانش میشود. اما اگر ریشه درخت خاموش باشد تعداد مینیمم برای آن میشود $1 + \sum_{i=1}^{r} min(Alg(T'_i))$ زیرا با روشن کردن این چراغ حالت تمام زیر درخت ها قرینه میشود. حالا میدانیم الگوریتم باز هم روی زیر درخت های جدید مینیمم روشن شدن آن ها را پیدا میکند و از آنجایی که برای روشن کردن ریشه طبق ادعا ۱ حتما یک بار یکی از اجداد این زیر درخت ها باید سوییچ میشد پس این ها حتما یک بار قرینه میشدند. در نتیجه باز هم مینیمم به دست آمده همان مینیمم خواسته شده است.

تحلیل زمانی:
این الگوریتم از یک BFS که الگوریتم چند جمله ایست استفاده میکند و سپس روی راس ها حرکت کرده و هر بار تعداد چند جمله ای چراغ را سوییچ میکند که در کل چند جمله ای میشود.

پرسش ۲

گراف جهت دار بدون دور یا همان DAG به ما یک توپولوژیکال سورت میدهد که به این ترتیب عمل میکنیم:

۱. اولین راس در توپولوژیکال سورت را برداشته و اگر این راس به راسی یال داشت آن راس را برمیداریم و راس های مجاور آن راس را حذف سپس در گراف باقی مانده اگر باز هم راس اول با راس دیگری مجاور بود آن راس را نیز برداشته و دوباره راس های مجاور آن را حذف میکنیم. (زیرا اگر این کار را انجام ندهیم مسیر به طول دو داریم)

۲. در بین راس های حذف نشده راس بعدی در توپولوژیکال سورت را انتخاب و دوباره مرحله قبل را روی آن اجرا میکنیم.

۳. پس از اینکه همه راس ها یا انتخاب شدند یا حذف الگوریتم را متوقف میکنیم.

اثبات درستی:

ادعا ۱: الگوریتم مسیر جهت دار به طول دو ندارد.

اثبات:
یکی از راس های انتخاب شده را در نظر میگیریم این راس اگر به راس دیگری یال داشته باشد آن راس دیگر به هیچ راسی یال ندارد.(زیرا اینگونه راس ها را حذف کردیم) و درجه ورودی این راس نیز صفر است زیرا اگر درجه ورودی داشت پس از اینکه این راس انتخاب میشد در فرایند الگوریتم راسی که به آن یال دارد حذف میشد . به همین ترتیب اگر راس انتخابی ما درجه خروجی نداشته باشد. و درجه ورودی داشته باشد راسی که از آن به این راس درجه ورودی هست درجه ورودی ندارد. پس مسیر به طول دو نداریم.

ادعا ۲: گراف خروجی حداقل $\frac{3n}{7}$ راس دارد.

اثبات:
در روند الگوریتم راسی که داریم روی آن فرایند حذف یا انتخاب را انجام میدهیم در نظر بگیرید.
اگر درجه خروجی این راس ۰ باشد یک راس انتخاب شده و هیچ راسی به ازای آن حذف نمیشود.
اگر درجه خروجی این راس ۱ باشد دو راس انتخاب شده و حداکثر دو راس حذق میشود.
اگر درجه خروجی آن ۲ باشد دو راس انتخاب شده و حداکثر چهار راس حذف میشود.
پس در بدترین حالت در هر مرحله اگر در نظر بگیریم ۳ راس انتخاب شده و ۴ راس حذف نسبت انتخاب شده ها به کل راس ها مینیمم $\frac{3}{7}$ می باشد که ادعا ثابت میشود.

پرسش ۳

ابتدا یک بار MST گراف داده شده را پیدا کرده و وزن آن را ذخیره میکنیم . سپس برای یک یال دلخواه نشان میدهم که در کدام یک از وضعیت های خواسته شده سوال قرار دارد.
تست ۱:
یال داده شده را حذف میکنیم روی گراف باقی مانده MST را پیدا میکنیم اگر وزن MST جدید بیشتر از MST گراف قبلی بود یال داده شده حتما در تمام MST ها حضور داشته.
تست ۲:
اگر نتیجه تست ۱ برای یک یال منفی بود یعنی این یال یا در هیچ کدام از MST ها نیست یا در بعضی از آنها هست. برای این کار یال داده شده را انتخاب و دو راس آن را در یک مجموعه قرار داده و الگوریتم prim را روی گراف اجرا میکنیم در نهایت اگر زیر درخت بدست آمده مجموع وزن برابر با MST گراف اصلی داشت آن نیز یک MST است و این یعنی آن یال در برخی از MST ها قرار دارد. در غیر این صورت یعنی ساخت MST با یال مورد نظر امکان ندارد و یال داده شده در هیچ یک از MST های گراف مورد نظر نیست.

اثبات تست ۱:
فرض کنیم که MST دیگری در گراف داده شده وجود دارد که شامل یال مورد نظر نیست در نتیجه اگر این یال را حذف کرده و MST را پیدا کنیم و وزن بیشتری داشته باشد به این معنیست که هیچ MST با وزن کمتر در گراف داده شده بدون این یال وجود نداشته در نتیجه حضور این یال وزن MST را کم کرده در نتیجه این یال حتما در تمام MST ها هست.

اثبات تست ۲:
الگوریتم پریم دو مجموعه دارد که فرض میکند مجموعه که تا کنون انتخاب شده بخشی از یک MST است و طبق این فرض MST را کامل میکند. حالا ما با اینکار فرض کردیم این یال بخشی از یک MST است و سعی کردیم آن را کامل کنیم اگر توانستیم که یعنی MST وجود داشته که شامل این یال باشد و اگر نتوانستیم یا وزن آن بیشتر شد یعنی فرض اولیه اشتباه بوده و این یال بخشی از یک MST نبوده است.

پرسش ۴

پرسش ۵

برای حل این سوال ابتدا سه بار الگوریتم دایکسترا را از راس های w و v و u اجرا کرده و به ازای هر راس مینیمم فاصله آن تا هر سه راس مد نظر را داریم.
حالا به صورت زیر عمل میکنیم:
روی تمام راس ها پیمایش کرده و مجموع فاصله ها از u و v و w را با هم جمع کرده و از آنها مینیمم میگیریم. سپس راسی که به عنوان مینیمم مجموع فاصله های آن خروجی داده میشود را بر میداریم و دوباره دایکسترا را روی آن اجرا میکنیم این بار مقادیر pre را هم برای مشخص کردن مسیر نگه میداریم. و سپس یال های این مسیر ها برای آن راس تا راس های u و سپس v و سپس w به زیر گراف خروجی اضافه میکنیم.

اثبات درستی:

ممکن است برای هر راس عدد داده شده دقیقا مجموع فاصله آن راس از u و v و w نباشد (مثلا مسیر آن راس به

u زیر گرافی از مسیر آن راس به v باشد). اما حتما یک راس وجود دارد که دقیقا با جمع کردن این سه مقدار اندازه مسیر مینیمم را به ما میدهد.
فرض کنیم زیرگرافی با مجموع وزن مینیمم داریم. قطعا برای یکی از راس های این گراف سه مسیر آن راس به u و v و w هیچ اشتراکی با هم ندارند و وقتی که الگوریتم فاصله این سه راس از راس مورد نظر در الگوریتم محاسبه میشود وزن این زیر گراف اضافه شده و این زیر گراف یا زیر گرافی هم وزن آن به عنوان خروجی انتخاب میشوند.

تحلیل زمان اجرا:

ابتدا سه بار دایکسترا را اجرا میکنیم که $O(m \log n)$.
سپس روی راس ها پیمایش کرده و سه عدد را جمع میکنیم$O(n)$.
در نهایت ماکسیمم اعداد بدست آمده در مرحله قبل را محاسبه میکنیم $O(n)$. و در آخر یک بار دیگر دایکسترا را اجرا میکنیم که در مرحله قبل محاسبه شده و یال های بدست آمده را نیز درج میکنیم.
که در مجموع $O((m+n)\log n)$ است.

پرسش ۶

برای حل این سوال یک گراف جهت دار کمکی میسازیم به این صورت که ابتدا روی هر راس دایکسترا میزنیم و فاصله آن راس از راس های دیگر به دست می آید و حالا به راس هایی که فاصله آنها کمتر مساوی طولی که تاکسی میتواند طی کند است میتوان با آن تاکسی رفت. پس در گراف جدید راسی که الان روی آن هستیم را در نظر میگیریم و از آن راس به راس هایی که میتوان با آن تاکسی رفت یک یال جهت دار به وزن هزینه تاکسی رسم میکنیم به ازای همه راس ها اینکار را انجام میدهیم. حالا در گراف جدید هزینه های مختلفی که از هر راس میتوان به راس دیگر رفت به حالت های مختلف را داریم کافیست روی گراف جدید دایکسترا از s بزنیم و مینیمم فاصله آن تا t را محاسبه کنیم. که همان مینیمم هزینه است زیرا وزن های گراف جدید بر اساس هزینه هاست.
تحلیل زمانی الگوریتم:
روی هر راس یک دایکسترا اجرا کردیم که اردر هر دایکسترا $O(m \log n)$ است. پس حالا که n بار آن را اجرا کردیم داریم:
$O(nm \log n)$.
حالا در گراف جدید اگر حتی گراف کامل باشد $O(n^2)$ یال داریم که این یعنی اجرا دایکسترا روی آن $O(n^2 \log n)$ است.
در نتیجه زمان اجرای کل الگوریتم برابر است با $O((n^2+mn)\log n)$.

اثبات درستی:
درستی این الگوریتم به وضوح اثبات میشود به این صورت که به ازای هر تاکسی تمام مسیر هایی که میتوان با آن تاکسی طی کرد به صورت جهت دار مشخص شده و هزینه آن نیز الحاق شده و درنهایت تمام مسیر های قابل طی با هزینه های آنها در نظر گرفته شده و با اجرای دایکسترا روی آن گراف روی راس s کوتاه ترین مسیر از لحاظ هزینه ای مشخص میشود.