دانشکده‌ی علوم ریاضی

آنالیز الگوریتم ها	مدرس: دکتر شهرام خزایی

تمرین سری دو

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

فرض ۱: الگوریتمی داریم که در زمان خطی مولفه های قویا همبند را در گراف جهت دار پیدا میکند.
فرض ۲: الگوریتمی داریم که در زمان خطی sort topological را برای یک dag پیدا میکند. در ادامه این الگوریتم هارا ارائه میدهم.

پرسش ۱

پرسش ۲

با گرفتن لیست اسکرین شات نفر اول ترتیب تمام افراد به جز نفر اول را داریم گراف جهت داری از $O(n)$ میسازیم به صورتی که نفر i به نفر j به صورت جهت دار متصل میشود اگر نفر i در لیست فعالیت ها بالا تر باشد. حالا از روی لیست نفر دوم اسم نفر اول را پیدا کرده و بین دو نفری که قرار دارد یال های جهت دار را رسم میکنیم در بدترین حالت دو راس نفر اول و نفر i ام هر دو بین دو نفر j و k قرار میگیرند که در اسکرین شات سوم ترتیب نفر اول و i نیز مشخص میشود. در کل در بدترین حالت با سه اسکرین شات لیست افراد تشکیل میشود.(ممکن است با دو اسکرین شات نیز این لیست تشکیل شود) و دوباره از روی تمام اسکرین شات ها این ترتیب را چک میکنیم که دقیقا همان گرافی که ساختیم با غیبت یک راس که خود شخص فرستنده است باشد. و اگر گراف به صورت یکتا مشخص شده بود و اسکرین شات ها تناقضی نداشتند به سادگی با پیمایش روی گراف ساخته شده اسامی را خروجی میدهیم و در غیر این صورت میگوییم هیچ ترتیب زمانی وجود ندارد. زمان این الگوریتم به وضوح $O(kn)$ است.(k اسکرین شات داریم و پیمایش و چک کردن هر کدام $O(n)$ زمان میبرد).

پرسش ۳

ابتدا از روی لیست f_i ها گرافی جهت دار که راس های آن کارمندان و یال های آن به صورت جهت دار از هر کارمند به کارمندی است که باید به آن خبر رسانی کند تشکیل میدهیم. سپس با الگوریتم 1 مولفه های قویا همبند را $O(|V|+|E|)$ پیدا میکنیم که در اینجا تعداد یال ها دقیقا با تعداد راس ها برابر است (زیرا هر کارمندی دقیقا به یک کارمند یال جهت دار دارد.) پس $O(n)$.
ادعا:
هر شخصی خبری که میشنود به همه افراد منتقل میشود اگر و تنها اگر تمام افراد در یک مولفه قویا همبند قرار گیرند.
اثبات :
اگر تمام افراد در یک مولفه قویا همبند قرار گیرند به وضوح از هر راس (کارمند) به راس های دیگر مسیری جهت دار هست پس خبر به خوبی پخش میشود.

حال میخواهم نشان دهم اگر تمام افراد در یک مولفه قویا همبند نباشند فردی وجود دارد که اگر خبر به او برسد همه مطلع نمیشوند. می دانیم هر dag حداقل یک sink دارد پس اگر خبر به یکی از افراد در مولفه ای که sink است برسد خبر فقط در همان مولفه میماند و افراد در مولفه های دیگر از آن خبر مطلع نمیشوند پس ادعا ثابت میشود.
ادعا:
کمترین یال مورد نیاز برای همبند کردن گراف dag برابر است با $max(a, b)$ که a تعداد رأس ها با درجه ورودی صفر و b تعداد رأس ها با درجه خروجی صفر است.
اثبات:
اگر از هر رأس به رأس دیگر راه وجود داشته باشد حتما تمامی رأس ها حداقل یک یال ورودی و خروجی دارند. پس حداقل به تعداد $max(a, b)$ نیاز داریم. حالا الگوریتمی ارائه میدهم که با این تعداد dag را تبدیل به گراف قویا همبند کند. توجه کنید که راس هایی که یا درجه ورودی ندارند یا درجه خروجی یا سینک هستند و یا سورس . ابتدا مشخص میکنیم که a بیشتر است یا b فرض کنیم که a>b داریم:
سورس ها بیشتر از سینک ها در نتیجه هر سینک را به سورس بعدی آن وصل میکنیم برای ایجاد دور همیلتونی و اگر دو یا چند سورس بعد از سینک مورد نظر بود سینک مورد نظر را به هر دو وصل میکنیم به صورت شهودی مانند قطعه های پازل گراف های dag را میچینیم و سینک های قطعه قبلی را به سورس های قطعه بعدی متصل میکنیم. با اینکار به وضوح دور همیلتونی تشکیل میشود و تعداد یال های مورد نیاز به تعداد سورس هاست. اگر سینک ها بیشتر بود یعنی b>a نیز به صورت مشابه عمل میکنیم و الگوریتمی ارائه دادیم که با $max(a, b)$ دور همیلتونی ایجاد میکند و اثبات کامل میشود.

پرسش ۴

اشتباهه ممکنه چنتا سینک داشته باشم. طبق الگوریتم ۱ ابتدا مولفه های قویا همبند را برای شهر ها و مسیر هایی که از قبل بین آنهاست پیدا میکنیم. سپس آنها را sort topological میکنیم ممکن است در sort topological ما مولفه های همبندی ای وجود داشته باشند که به هیچ وجه به هم راه ندارند و اصلا به هم وابسته نیستند این مولفه ها را در دسته های جداگانه ای قرار میدهیم سپس برای هر sink u_i آن را در بخشی که هست پیدا میکنیم و برای هر sourse v_i آن را. حالا اگر u_i و v_i به هم مسیر جهت دار نداشتند یال جهت داری از سینک u_i به سورس v_i وصل میکنیم. این کار باعث میشود مسیر های جهت داری بین u_j هایی که از u_i به آنها راه هست به v_k هایی که در مسیر رسیدن به v_i قرار دارند ایجاد شود. این کار ممکن است کار مارا برای ساخت مسیر کم کند. حالا کافیست درستی الگوریتم را اثبات و زمان آن را تحلیل کنیم.
ادعا:
این الگوریتم به ازای هر i v_i را به v_j مسیر دار میکند.
اثبات:
در هر مرحله با اجرای DFS مشخص میشود که آیا مسیری بین این دو راس هست یا خیر و اگر نباشد به سادگی با وصل کردن سینک u_i به سورس v_i این مسیر ساخته میشود.
ادعا:
این الگوریتم کمترین تعداد مسیر مورد نیاز را به ما میدهد.
اثبات:
برای اثبات این ادعا باید دو چیز را نشان دهیم یکی اینکه اگر در مرحله i ام بین دو شهر u_i و v_i مسیری جهت دار نباشد حداقل یک مسیر جدید باید ایجاد شود برای اتصال این دو به هم و مورد بعد این است که نشان دهیم با هر جایگشتی از u_i ها تعداد مسیر های به دست آمده یکسان است. مورد اول بدیهی است پس کافیست نشان دهیم که با هر ترتیبی u_i ها را انتخاب و با الگوریتم داده شده مسیری از آن به v_i ایجاد کنیم تفاوتی در تعداد مسیر ها ایجاد

نمیشود. برای اینکار کافیست نشان دهیم در یک جایگشت داده شده با عوض کردن جای دو عنصر دلخواه u_i و u_j تفاوتی ایجاد نمیشود. چون میدانیم از هر جاگشتی با عوض کردن دو تا دوتا عناصر میتوان به هر جایگشت دلخواه رسید.
عناصر را به سه دسته قبل از u_i و بین u_i و u_j و بعد از u_jتقسیم میکنیم .
جایگشت اول به صورت $u_1, ..., u_i, ..., u_j, ..., u_m$ و جایگشت دوم به صورت $u_1, ..., u_j, ..., u_i, ..., u_m$ است.
میتوان جایگشت ها را به صورت زیر نگاه کرد $u_i, ..., u_j, ..., u_m$ و $u_j, ..., u_i, ..., u_m$ زیرا قبل از آن دقیقا تغییرات یکسانی را در گراف ایجاد کرده چهار حالت را بررسی میکنم:
۱: u_i و u_j به ترتیب در جایگشت های اول و دوم مسیری اضافه نکنند: در این حالت وقتی u_i در جایگشت اول مسیر ایجاد نکرده یعنی تا قبل از آن مرحله به v_i متناظر خود وصل شده پس در جایگشت دوم نیز لزومی به اتصال ندارد و همچنین برای u_j. پس این دو عنصر از جایگشت بی اثر میشوند و هر دو جایگشت یکی است.
۲:u_i در جایگشت اول مسیر ایجاد کند و u_j در جایگشت دوم مسیر ایجاد نکند: در این حالت در جایگشت اول مسیر ایجاد شده توسط u_i باعث بی اثر شدن عناصری در جایگشت شده که سینک آنها با سینک u_i و سورس آنها با سورس v_i یکی بوده است. درنتیجه در جایگشت دوم اگر یکی از این عناصر زودتر از u_i ظاهر شود u_i و همه آن عناصر را بی اثر کرده و تعداد مسیر ها ثابت میماند. برای u_j چون در جایگشت دوم بی اثر است در جایگشت اول نیز (مانند استدلال قسمت قبل) بی اثر است. در نتیجه دو حالت دیگر که به صورت زیر هستند نیز با همین استدلال ثابت میشوند.
۳:u_j در جایگشت اول مسیر ایجاد کند و u_i در جایگشت دوم مسیر ایجاد نکند.
۴:u_i و u_j به ترتیب در جایگشت های اول و دوم مسیری اضافه کنند.
پس ادعا ثابت میشود .
تحلیل زمانی:
در هر مرحله برای پیدا کردن مولفه های قویا همبند و sort topological به $O(n)$ زمان نیاز داریم (که n مجموع راس و یال های گراف است.)
و اما در نهایت برای m راس باید این کار را انجام دهیم پس $O(nm)$ زمان مورد نیاز برای اجرای این الگوریتم میباشد.

پرسش ۵

میخواهیم برای هر راس

پرسش ۶

برای اثبات این الگوریتم کافیست دو چیز را ثابت کنیم که به صورت دو ادعا در زیر ارائه میکنم.

در ابتدا به این نکته توجه میکنیم که در درخت هر گاه مسیری بین دو رأس داریم آن مسیر تنها مسیر بین آن دو رأس است. (زیرا در غیر این صورت دور تشکیل میشود) درنتیجه مسیری که بین دو رأس i و j در درخت BFS وجود دارد ابتدا از i به ریشه میرویم و اگر j در این مسیر نبود از j به سمت ریشه میرویم و اولین جایی که به مسیر از i به ریشه برخورد کردیم میتوانیم به سمت i برویم. پس هر مسیری یک بخش بالا رونده به سمت ریشه و یک بخش پایین رونده به سمت رأس مورد نظر دارد. و نکته دوم قابل توجه این است که قطر درخت هرگز نمیتواند بین i و j ای باشد که یکی در مسیر دیگری به ریشه است. زیرا فاصله آن رأس تا ریشه بیشتر از فاصله آن تا رأس دیگر میشود.

ادعا ۱:
اگر قطر گراف طول D داشته باشد هر رأس را به عنوان ریشه درخت BFS انتخاب کنیم حتما یک سر قطر در سطح آخر می افتد.
اثبات:
رأس v را به عنوان ریشه درخت BFS در نظر میگیریم. سر اول قطر از آن فاصله H و سر دوم قطر از آن فاصله $D-H$ دارد. در این صورت ارتفاع درخت BFS برابر با $max(H, D-H)$ میشود. زیرا اگر بیشتر شود رأسی به فاصله بیشتر از این دو، از ریشه درخت وجود دارد که این یعنی آن رأس به عنوان سر دیگر قطر شناخته میشود.

ادعا ۲:
اگر در درخت BFS یک راس از سطح آخر انتخاب کنیم و BFS را روی آن اجرا کنیم مستقل از اینکه آن کدام رأس است عدد یکسانی به عنوان قطر به ما میدهد.
اثبات:
فرض کنیم $a_1, a_2, .., a_n$ رأس های سطح آخر ما هستند. اگر a_i راس i را به عنوان سر دیگر قطر خروجی دهد اگر a_j را نیز در نظر بگیریم . یا جد مشترک آن با i همان جد مشترک a_i با i است که در این صورت همان فاصله را با i دارد. و یا جد مشترک آن با i متفاوت از جد مشترک a_i و i است که در این صورت اگر در نظر بگیریم جد مشترک a_j و i پایین تر از جد مشترک a_i و i است. اگر i در سطح آخر نباشد آنگاه از جد مشترک i و a_j بجای رفتن به i به a_j میرویم. که این یعنی طول قطر برای a_i بیشتر میشد. پس اگر جد مشترک متفاوت داشتیم نیز فاصله a_j و a_i با هم را به عنوان خروجی برای a_j در نظر میگیریم. (زیرا برابر با فاصله a_i و i بود.) حالا کافیست نشان دهیم که هیچ کدام از a_i ها نمیتوانند عدد بزرگ تری از دیگری به قطر نسبت بدهند. این بخش نیز به وضوح معلوم است چون نشان دادیم هر عددی که از هر کدام از a_i ها بگیریم به سادگی میتوان به a_j دیگر نیز راسی نسبت داد که همان عدد را بدهد. پس همگی مساوی و ماکسیمم نداریم.

و اثبات کامل میشود.