پاسخ امتحان چهارم - np

طراحی الگوریتم - پاییز ۹۹

سوال ۱:

الف) درست است. طبق تعریف مسائل np-hard، اگر A در زمان چند جمله ای حل شود آنگاه تمام مسائل np نیز در زمان چند جمله ای حل خواهند شد. چون A را نیز می توان در زمان چند جمله ای به B کاهش داد پس اگر حل شود نیز ابتدا A و سپس همه مسائل np حل می شوند. پس B نیز np-hard است.

ب) درست است. اگر یکی از مسائل np-complete حل شود تمام مسائل np و به طبع مسائل np-complete حل خواهند شد.

ج) درست است. مسائلی np-complete هستند که ابتدا np باشند و دوما شرط np-hard بودن را داشته باشند. پس np-complete زیرمجموعه np-hard است.

د) نادرست است. در دسته بندی مسائل حرفی از حل با زمان خطی زده نمی شود. صرفا اینکه در زمان چند جمله ای حل شود کافی است.

سوال ۲:

الف) درست است. شرط اول مسائل np-complete این است که np باشند پس بدیهتا زیر مجموعه مسائل np هستند

ب) نادرست است. ممکن است B مسئله np-hard باشد و A چون به آن کاهش می یابد با حل B

A و تمام مسائل np شوند. ولی اینکه A وریفایر دارد دلیل نمی‌شود که B نیز داشته باشد تا np باشد

ج) نادرست است. می دانیم اگر یکی از مسائل np-hard حل شود همه مسائل np حل خواهند شد و

هیچ شرطی بیشتری نداریم. ولی همه مسائل np-hard لزوما np نیستند و ممکن است حل نشوند.

د) نادرست است. این شرط مخصوص مسائل np-complete است. مسائل p نیز داخل np هستند و

می دانیم در زمان چند جمله ای حل می شوند پس حل آن ها چیز جدیدی نخواهد بود.

سوال ۳:

برای این که حکم را ثابت کنیم، باید نشان بدهیم اگر مسئله‌ی SET-COVER را حل شده داشته باشیم، آن‌گاه با زمانی چند جمله‌ای می‌توانیم مسئله‌ی VERTEX-COVER را نیز حل کنیم. ادعا می‌کنیم می‌توانیم این کار را انجام دهیم.

اثبات ادعا: یک مسئله‌ی VERTEX-COVER را در نظر بگیرید. پس گراف G و عدد k داریم که می‌خواهیم بدانیم آیا k رأسی وجود دارد که با رنگ کردن آن‌ها هر یال گراف حداقل یکی از دو سر آن رنگ شده باشد. یال‌های این گراف را از 1 تا m به دل‌خواه شماره گذاری می‌کنیم. حال برای کاهش مسئله، U را برابر با مجموعه‌ی ۱ تا m در نظر می‌گیریم. هر عضو $\mathcal{A}$ را نیز به این شکل انتخاب می‌کنیم که به ازای هر رأس گراف G، مجموعه یال‌هایی که به این رأس وصل هستند را به $\mathcal{A}$ اضافه می‌کنیم. می‌دانیم که این کار از مرتبه‌ی زمانی تعداد یال‌های گراف است زیرا که هر یال تنها دو بار دیده می‌شود. حال ادعا می‌کنیم جواب مسئله‌ی SET-COVER با پارامترهای ساخته شده‌ی U، $\mathcal{A}$

و k، معادل با جواب مسئله است. توجه شود که معادل بودن شرطی دو طرفه است پس نیاز دارد که هر دو طرف اثبات شود.

اثبات طرف اول: اگر SET-COVER جواب را بله بگوید، جواب VERTEX-COVER نیز بله خواهد بود؛ زیرا از آن جایی که جواب SET-COVER بله بوده است، پس وجود دارد k عضو از مجموعه‌ی $\mathcal{A}$ که مجموعه‌ی U را پوشش بدهند. حال در گراف G رئوس معادل این مجموعه‌ها را نگاه کنید. از آن‌جایی که این مجموعه‌های کل U را پوشش می‌دهند، پس همه‌ی یال‌های گراف G حداقل یک سر آن‌ها یکی از این رئوس انتخاب شده است.

اثبات طرف دوم: اگر VERTEX-COVER جوابش بله باشد، حتما جواب SET-COVER ساخته شده نیز بله خواهد بود؛ زیرا از آن جایی که جواب VERTEX-COVER بله بوده است، پس k رأس وجود دارد که شرایط مسئله را برقرار کنند. حال مجموعه‌های معادل این رئوس را در $\mathcal{A}$ در نظر بگیرید. انتخاب این مجموعه‌ها (که تعدادشان دقیقا k است) باعث پوشاندن مجموعه‌ی U می‌شود زیرا که یال‌های گراف G با این رئوس پوشانده شده بود.

سوال ۴:

برای این که حکم را ثابت کنیم، باید نشان بدهیم اگر مسئله‌ی KNAPSACK را حل شده داشته باشیم، آن‌گاه با زمانی چند جمله‌ای می‌توانیم مسئله‌ی SUBSET-SUM را نیز حل کنیم. ادعا می‌کنیم می‌توانیم این کار را انجام دهیم.

اثبات ادعا: یک مسئله‌ی SUBSET-SUM را در نظر بگیرید. مجموعه‌ی A از اعداد طبیعی و عدد k را داریم که می‌خواهیم بدانیم زیرمجموعه‌ای از A وجود دارد که جمع اعداد آن k بشود یا نه. حال برای کاهش این مسئله، پارامترهای مسئله‌ی KNAPSACK را تعیین می‌کنیم. حجم کوله‌پشتی یعنی V را برابر با k و هم‌چنین پارامتر k ی مسئله‌ی KNAPSACK را نیز برابر با k ی مسئله‌ی SUBSET-SUM قرار می‌دهیم. سپس برای هر عدد از مجموعه‌ی A مثل x، یک کالا با حجم و ارزشی برابر با x قرار می‌دهیم. از آن جایی که فقط یک بار تمام اعداد را دیده‌ایم، پس معادل کردن این مسئله در زمان چند جمله‌ای اتفاق افتاده است. حال ادعا می‌کنیم که جواب مسئله‌ی SUBSET-SUM، معادل با جواب این مسئله از KNAPSACK خواهد بود. برای اثبات این ادعا ابتدا یک طرف این ادعا و سپس طرف دیگر آن را اثبات می‌کنیم.

اثبات طرف اول: اگر جواب مسئله‌ی KNAPSACK ما بله باشد، جواب مسئله‌ی SUBSET-SUM نیز بله خواهد بود؛ زیرا از آن جایی که جواب KNAPSACK بله است، پس زیرمجموعه‌ای از کالاها وجود دارد که مجموع حجم آن‌ها کم‌تر مساوی k و مجموع ارزش آن‌ها بیش‌تر مساوی k شود. اما از آن‌جایی که حجم و وزن کالاها با هم برابر است، پس می‌توان نتیجه گرفت که هم حجم زیرمجموعه و هم ارزش آن برابر با k است. حال کافی است که از A در مسئله‌ی SUBSET-SUM اعداد معادل با این زیرمجموعه‌ی مشخص شده را انتخاب کنیم. از آن جایی که ارزش و حجم این زیرمجموعه برابر با k شده، پس جمع اعداد انتخاب شده قطعا k خواهد بود.

اثبات طرف دوم: اگر جواب مسئله‌ی SUBSET-SUM بله باشد، حتما جواب KNAPSACK ساخته شده بله خواهد بود؛ زیرا کالاهای معادل با اعدادی از مجموعه‌ی A که جمع آن‌ها k است را

در نظر بگیرید، از آن جایی که وزن و حجم این کالاها معادل با مقدار آن‌ها در A است، پس ارزش و حجم این کالاها k خواهد بود که شرایط مسئله‌ی KNAPSACK را برآورده می‌کند.

سوال ۵:

الف) برای اینکه ثابت کنیم یک مسئله در کلاس np است باید نشان دهیم برای راه حل آن وریفایری در زمان چند جمله ای وجود دارد. در این مسئله اگر k راس به ما خروجی داده شود و ادعا شود که هیچ مثلثی در آن وجود ندارد می توان در وریفای به ازای هر ۳ راسی (با مرتبه $O(n^3)$) چک کنیم که اگر سه تا یال ممکن بینشان وجود داشته باشد مثلث است و ادعا را رد کنیم.

ب) مسئله Independent-set را به مسئله triangle-free کاهش می دهیم. در ورژن yes/no سوال مجموعه مستقل گراف G و عدد k را به عنوان ورودی داریم. باید مشخص کنیم این گراف k راس دارد که مستقل باشند. (هیچ یالی بینشان نباشد) ورودی های آن را به ورودی مسئله تبدیل و از خروجی آن خروجی اش را می سازیم. از روی گراف G گراف 'G را می سازیم. به آن n راس (تعداد رئوس G) اضافه می کنیم. همه آن ها را به همه n راس اصلی متصل می کنیم. ادعا می کنیم وجود مجموعه ای n+k راسی خالی از مثلث در 'G متناظر وجود مجموعه مسقل k راسی در G است. این حکمی دو طرفه است که دو طرف آن باید اثبات شود. طرف اول: اگر مجموعه n+k راسی وجود داشته باشد، آنگاه حداقل k راس از آن داخل n راس اصلی است. بین این k راس نباید هیچ یالی وجود داشته باشد؛ زیرا اگر باشد با یکی از رئوس اضافی ایجاد مثلث می کنند. حال در طرف دوم داریم: اگر مجموعه بدون مثلث n+k تایی نداشته باشیم، مجموعه k تایی مستقل در راس های اصلی نخواهیم داشت.

فرض خلف می کنیم که وجود داشته باشد؛ آنگاه این k راس و n راسی که اضافه کرده ایم n+k راس می سازند که چون در دو بخش یالی نداریم پس مثلثی ندارد. که این خلاف فرضمان است و تناقض است. پس کاهش ثابت شده است.

سوال ۶:

الف) باید نشان دهیم اگر خروجی مسئله FANCY-SET را که افزار مجموعه S به دو مجموعه S_1 و S_2 است داشته باشیم می توانیم آن را وریفای کنیم. برای هر کدام از زیرمجموعه های C چک می کنیم که آیا در هر کدام از S_1 و S_2 عضوی دارد یا نه. واضح است که چنین کاری زمان چندجمله ای دارد.

ب) باید مسئله 3-SAT را به FANCY-SET کاهش دهیم. n متغیر و m معادله به عنوان ورودی داریم که باید با مقداردهی درست یا غلط به هر متغیر معادله ها را ارضا کنیم. (حداقل یکی از ترم هایش برقرار باشد) ورودی ها را به ورودی FANCY_SET تبدیل می کنیم و از خروجی آن خروجی مطلوب را می سازیم. مجموعه S را شامل 2n+1 عنصر تشکیل می دهیم. 2n عنصر $x_1, x_2, \ldots, x_n$ و $\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n}$ متناظر هر متغیر و نات آن و یک متغیر f. (در ادامه فایده آن را متوجه می شویم) برای هر معادله نیز اعضای آن (اگر خود متغیر یا نات آن است) و متغیر f را داخل یک زیرمجموعه در C می گذاریم. همچنین برای هر متغیر خودش و نات آن ($x_i, \overline{x_i}$) آن را نیز در یک زیرمجموعه در C می گذاریم. (احتمالا متوجه شده اید که این کار برای آن است که جفت آن ها در یک مجموعه قرار نگیرند!) پس در کل C شامل n+m زیرمجموعه می شود.

مسئله FANCY_SET مجموعه S را به دو قسمت S_1 و S_2 تقسیم می کند. آن قسمتی که شامل متغیر f است را مجموعه غلط و مجموعه دیگر را مجموعه صحیح در نظر می‌گیریم. برای هر متغیر دقیقا یکی از خودش یا ناتش در مجموعه درست خواهد بود؛ اگر خودش باید آن را صحیح مقدار دهی کنیم و اگر نه غلط. می دانیم در هر زیر مجموعه حداقل یک عضو باید داخل S_1 باشد و حداقل یک عضو داخل S_2 . چون یک f در هر زیرمجموعه وجود دارد که متعلق به دسته اشتباه است پس حتما یک متغیر هم در دسته صحیح خواهیم داشت که همان آن معادله را درست خواهد کرد.

تا اینجا ثابت کردیم اگر FANCY_SET جواب داشته باشد آنگاه 3-SAT نیز جواب دارد. حال باید نشان دهیم اگر FANCY_SET جواب نداشته باشد آنگاه 3-SAT جواب ندارد. برهان خلف می زنیم. اگر جوابی داشته باشد؛ اگر طبق مقدار دهی دسته صحیح و غلط را تقسیم کنیم در نتیجه هیچ زیرمجموعه ای زیرمجموعه S_1 یا S_2 نیست. (عضو f و یک متغیر ارضا شده در مجموعه صحیح وجود دارد) پس FANCY_SET نیز جواب دارد. پس درستی کاهش اثبات شده است.