به نام خدا

# نظریه زبان‌ها و ماشین‌ها- بهار ۱۴۰۱

پاسخنامه‌ی تمرین شماره ۱۲

دستیار آموزشی این مجموعه: سید پارسا حسینی‌نژاد

hoseininejad1999@gmail.com

تاریخ تحویل : ۳۱ خرداد (صفحه درس)

---

1) **گراف‌های دوری گراف‌هایی هستند که یال‌های آن‌ها تشکیل تنها یک دور می‌دهند. زبان L، مجموعه گراف‌های غیر جهت‌داری مانند G است که در بر گیرنده‌ی حداقل یک گراف دوری با تعداد رأس‌های چهار هستند. با توجه به این تعریف، کدام گزاره دقیق‌تر است؟ $L \in P$ یا $L \in NP$؟ ادعای خود را ثابت کنید. (۱۰ نمره)**

<u>پاسخ</u>

با استفاده از الگوریتم زیر L را تصمیم‌گیری می‌کنیم:

```
On input graph G:
    For each permutation of four vertices (a, b, c, d):
        Return True if G contains edges (a, b), (b, c), (c, d), (d, a) and doesn't
        contain edges (a, c), (b, d)
    Return Flase
```

الگوریتم داده شده True برمی‌گرداند اگر و تنها اگر گراف G شامل یک گراف دوری ۴ رأسه باشد. تعداد رأس‌های گراف G را برابر n و تعداد یال‌های آن را برابر m در نظر می‌گیریم. در این صورت $O(n^4) = \frac{n!}{4!*(n-4)!}$ چهارتایی از رأس‌های موجود در گراف G است که ممکن است تشکیل یک گراف دوری بدهند. برای هر چهارتایی از رئوس بررسی وجود ۴ یال و وجود نداشتن ۲ یال در زمان $O(m)$ انجام می‌شود. بنابراین الگوریتم ارائه شده در زمان $O(mn^4)$ اجرا می‌شود. پس $L \in P$.

2) **الف) بسته بودن مجموعه زبان‌های دسته‌ی P را نسبت به عملیات‌های *(Star) و مکمل گیری نشان دهید. (۱۰ نمره)**
**ب) بسته بودن مجموعه زبان‌های دسته‌ی NP را نسبت به عملیات‌های *(Star) و اشتراک نشان دهید. (۱۰ نمره)**

<u>پاسخ</u>
الف) در بخش الف از زبان $L$ استفاده می‌کنیم و فرض می‌کنیم در دسته‌ی P قرار دارد.
<u>عملیات ستاره:</u> فرض کنید ورودی شما $x_1x_2...x_n$ باشد. حال از روش برنامه‌نویسی پویا استفاده می‌کنیم و آرایه‌ی $M$ را تعریف می‌کنیم به گونه‌ای که $M[i]$ در صورتی برابر True است که $x_1x_2...x_i$ در $L^*$ باشد و در غیر این صورت Flase است. ماشین تورینگ T را در نظر بگیرید که کارهای زیر را به ترتیب انجام می دهد:
۱- مقدار $M[0]$ را True بگذار، زیرا رشته‌ی خالی ϵ در $L^*$ است.
۲- تمامی مقادیر $M[i]$ را از ابتدا تا انتها به ترتیب محاسبه کن. اگر یک مقدار j با $0 \leq j < i$ وجود دارد که $A[j] = True$ (یعنی $x_1x_2...x_j \in L^*$) و $x_{j+1}x_{j+2}...x_i$ در L باشد، $A[i] = True$ قرار داده می‌شود. در غیر این صورت، $A[i] = Flase$ قرار داده می‌شود. این کار را برای تمامی اعداد i انجام می‌دهیم. این مرحله را می‌توان در زمان چند جمله‌ای انجام داد، زیرا می‌توانیم در زمان چند جمله‌ای تصمیم بگیریم که آیا $x_{j+1}x_{j+2}...x_i$ در L است یا نه (بر اساس فرضیه، زبان L در P است).
۳- اگر $A[n] = True$ است، قبول کن. در غیر این صورت رد کن.
پس دیدیم زبان $L^*$ نیز در دسته‌ی P قرار می‌گیرد.
<u>عملیات مکمل گیری:</u> فرض کردیم زبان $L$ در دسته‌ی P قرار دارد، حال باید ثابت کنیم $\overline{L}$ نیز در دسته‌ی P است. فرض می‌کنیم ماشین تورینگ مربوط به زبان $L$، ماشین $T$ است. حال ماشین $T'$ را اینگونه تعریف می‌کنیم: در صورتی که ماشین $T$ ورودی را پذیرفت، آن را رد کن و اگر ورودی را رد کرد، آن را بپذیر. از آنجایی که $T$ در زمان چندجمله‌ای کار می‌کند، پس $T'$ نیز در زمان چندجمله‌ای کار می‌کند و $\overline{L}$ نیز در دسته‌ی P خواهد بود.

ب) در بخش ب از دو زبان $L$۱ و $L2$ استفاده می‌کنیم و فرض می‌کنیم در دسته‌ی NP قرار دارند.
<u>عملیات ستاره:</u> فرض می‌کنیم ماشین تورینگ مربوط به زبان $L1$، ماشین $T$ است. حال ماشین $T'$ را اینگونه تعریف می‌کنیم:
۱- بررسی کن که آیا $w = \epsilon$. اگر هست، ورودی را قبول کن.

۲- در غیر این صورت، رشته‌ی $w$ را به صورت غیر قطعی (یعنی تصادفی) به تعدادی قطعه (هر تعدادی از ۱ تا $|w|$) تقسیم کن.
3. از $T$ استفاده کن تا ببینیم که آیا هر یک از قطعات در $L1$ هستند یا خیر. اگر همه هستند، قبول کن. اگر قطعه‌ای نیست، به قسمت ۲ برگرد و تقسیم جدیدی را انجام بده.
این ماشین در زمان چندجمله‌ای کار می‌کند، زیرا هم قسمت ۲ و هم قسمت ۳ در زبان چندجمله‌ای کار می‌کنند. پس زبان $L1^*$ نیز در دسته‌ی NP قرار می‌گیرد.
<u>اشتراک:</u> ماشین‌های تورینگ مربوط به هر دو زبان $L1$ و $L2$ در زمان چندجمله‌ای متوقف می‌شوند. حال ورودی جدید را ابتدا به ماشین تورینگ زبان $L1$ و سپس $L2$ می‌دهیم. اگر هر دو پذیرفتند، ورودی قبول می‌شود و در غیر این صورت رد می‌شود. پس می‌توان در زمان چندجمله‌ای در مورد ورودی جدید تصمیم‌گیری کرد، پس اشتراک دو زبان NP در NP است.

**3) زبان SAT-SAT با تعریف زیر را در نظر بگیرید:**

**SAT-SAT = {** $\langle \varphi \rangle$ **|**

**حداقل دو مقداردهی مختلف وجود دارد که به ازای آن‌ها عبارت منطقی φ برقرار می‌شود }**

**ثابت کنید SAT-SAT یک زبان NP-Complete است. (۲۰ نمره)**

<u>پاسخ</u>
ابتدا ثابت می‌کنیم SAT-SAT در دسته‌ی NP قرار می‌گیرد. به این منظور باید یک certificate ارائه دهیم. این certificate از دو مقداردهی متفاوت $a1$ و $a2$ تشکیل شده است. برای هر مقداردهی، مقادیر را با متغیرهای متناظر جایگزین می‌کنیم و بررسی می‌کنیم که آیا φ را برقرار می‌کند یا نه. مشخص است که این بررسی در زمان چندجمله‌ای صورت میگیرد، پس SAT-SAT در NP است.

حال باید ثابت کنیم این مسأله NP-HARD است. به این منظور SAT را به SAT-SAT کاهش می‌دهیم. به ازای ورودی $\varphi(x_1, x_2, \dots, x_n)$، متغیر جدید $y$ را معرفی می‌کنیم و فرمول خروجی را اینگونه تعریف می‌کنیم:

$$\varphi'(x_1, x_2, \dots, x_n, y) = \varphi(x_1, x_2, \dots, x_n) \wedge (y \vee \overline{y})$$

اگر $\varphi(x_1, x_2, \dots, x_n)$ به SAT تعلق داشته‌باشد، پس φ حداقل یک مقداردهی دارد که به ازای آن عبارت صحیح می‌شود. پس، 'φ حداقل ۲ مقداردهی درست خواهد داشت، زیرا به ازای هر

دو حالت $y = False$ یا $y = True$، یک مقداردهی درست برای φ خواهیم داشت. پس $SAT.SAT \ni \varphi'(x_1, x_2, \dots, x_n, y)$.

حال، اگر $\varphi(x_1, x_2, \dots, x_n)$ به SAT تعلق نداشته‌باشد، پس $\varphi'(x_1, x_2, \dots, x_n, y) = \varphi(x_1, x_2, \dots, x_n) \wedge (y \vee \bar{y})$ هم یک مقداردهی مناسب نخواهد داشت، پس $SAT.SAT \notin \varphi'(x_1, x_2, \dots, x_n, y)$.

پس توانستیم SAT را به SAT-SAT کاهش دهیم، پس SAT-SAT یک زبان NP-Complete است.

4) **فرض کنید مجموعه‌ی $M = \{x_1, x_2, \dots, x_n\}$ وجود دارد. S، یک مجموعه‌ی mتایی از زیرمجموعه‌های M است که اجتماع آن‌ها برابر مجموعه‌ی M می‌شود. برای مثال، اگر $M = \{1, 2, 3, 4, 5, 6\}$ باشد، S می‌تواند $S = \{\{1, 2\}, \{1, 2, 3\}, \{4\}, \{4, 5\}, \{5, 6\}\}$ باشد. یک K-SET از مجموعه‌ی S، تعداد k عضو از S است که اجتماع آن‌ها برابر M خواهد شد. برای مثال، یک $3 - SET$ از مجموعه‌ی S می‌تواند برابر $S = \{\{1, 2, 3\}, \{4, 5\}, \{5, 6\}\}$ باشد. ثابت کنید مسئله‌ی K-SET با ورودی‌های $\langle M, S, k\rangle$ یک مسئله‌ی NP-Complete است (می‌توانید از مسئله‌ی VERTEX-COVER استفاده کنید). (۲۰ نمره)**

<u>پاسخ</u>

ابتدا ثابت می‌کنیم K-SET در دسته‌ی NP قرار می‌گیرد. به این منظور باید یک certificate ارائه دهیم. به این منظور، اگر تعدادی عضو از مجموعه‌ی S با اندازه k ارائه شود، می‌توانیم روی هر عنصر در زیر مجموعه‌های مجموعه حرکت کنیم و عناصری از M را که پوشانده شده‌اند علامت‌گذاری کنیم. در پایان، هیچ عنصری در M نباید بدون علامت باقی بماند. این عملیات در زمان چندجمله‌ای صورت می‌گیرد. بنابراین K-SET در NP است.

حال ثابت می‌کنیم این زبان یک زبان NP-Hard است. به این منظور، از مسئله‌ی VERTEX-COVER که یک مسئله‌ی NP-Complete است استفاده می‌کنیم و آن را به مسئله‌ی K-SET کاهش می‌دهیم. به این منظور، فرض می‌کنیم یک گراف ورودی شامل تعدادی رأس و یال و یک عدد k داریم. با توجه به گراف ورودی، مجموعه‌ی M را می‌سازیم که شامل تمامی یال‌های گراف است. مجموعه‌ی S را هم به گونه‌ای تشکیل می‌دهیم که هر عضو آن مربوط به یال‌های خروجی از یک رأس خاص باشد.

حال اگر یک VERTEX-COVER kتایی وجود داشته باشد، یعنی می‌توان k رأس انتخاب کرد که تمامی یال‌ها را پوشش می‌دهند. پس می‌توان یک K-SET معرفی کرد، پس $(M, S, k) \in SAT.SAT$.

اگر یک VERTEX-COVER kتایی وجود نداشته باشد، یعنی نمی‌توان با حداقل k رأس تمامی یال‌ها را پوشش داد. پس نمی‌توان یک K-SET معرفی کرد، پس $(M, S, k) \notin SAT.SAT$.

پس ثابت شد مسئله‌ی K-SET یک مسئله‌ی NP-Complete است.

**5) زبان H-CLIQUE با تعریف زیر را در نظر بگیرید:**

**H-CLIQUE = { ⟨G⟩ |**

**G یک گراف بدون جهت است که دارای یک زیرگراف کامل با حداقل گره‌های $\left|\frac{m}{2}\right|$ است که m تعداد گره های G است }**

**ثابت کنید H-CLIQUE یک زبان NP-Complete است. (۲۰ نمره)**

<u>پاسخ</u>

این زبان NP است زیرا certificate مربوط به این زبان و زبان CLIQUE برابر است و می‌دانیم این زبان NP است.

حال ثابت می‌کنیم این زبان یک زبان NP-Hard است. به این منظور، از مسئله‌ی CLIQUE که یک مسئله‌ی NP-Complete است استفاده می‌کنیم و آن را به مسئله‌ی H-CLIQUE کاهش می‌دهیم. ورودی تابع کاهش دهنده، جفت ⟨G, k⟩ است و عملیات کاهش گراف ⟨H⟩ را خروجی می‌دهد. فرض کنید m تعداد گره‌ها در گراف G باشد. ۳ حالت زیر را در نظر می‌گیریم:

۱- اگر $k = \frac{m}{2}$، پس $H = G$.

۲- اگر $k > \frac{m}{2}$، برای تولید گراف H، مقدار t گره‌ی درجه صفر را به نمودار G اضافه می کنیم که در آن $t = 2k - m$. پس گراف H دارای تعداد کل گره‌ها معادل $m + t = 2k$ خواهد بود. حال، گراف G نیز یک k-clique خواهد داشت اگر و تنها اگر گراف H یک clique به اندازه‌ی k داشته باشد. پس، $\langle G,\ k\rangle \in CLIQUE$ اگر و تنها اگر $\langle H\rangle \in H.CLIQUE$.

۳- اگر $k < \frac{m}{2}$، برای تولید گراف H، مقدارt گره را به گراف G اضافه می‌کنیم و این گره‌ها را به تمامی گره‌های موجود در گراف با یک یال متصل می‌کنیم که در آن $t = m - 2k$. پس گراف H

دارای تعداد کل گره‌ها معادل $m + t = 2m - 2k$ خواهد بود. می‌بینیم که گراف G یک k-clique خواهد داشت اگر و تنها اگر H یک clique به اندازه‌ی $k + t = m - k$ داشته باشد. پس، $\langle G, k\rangle \in CLIQUE$ اگر و تنها اگر $\langle H\rangle \in H.CLIQUE$.

پس ثابت شد مسئله‌ی H-CLIQUE یک مسئله‌ی NP-Complete است.

**6) (امتیازی) مسئله‌ی زیر را در نظر بگیرید:**
**دانشکده‌ی برق و کامپیوتر دانشگاه تهران هر ترم n درس ارائه می‌دهد که هر کدام از این درس‌ها در بازه‌ی زمانی S برگزار می‌شوند و این بازه‌های زمانی می‌توانند با یکدیگر تداخل داشته باشند. همچنین می‌دانیم دانشگاه k کلاس دارد و $k < n$. آیا می‌توان درس‌ها را به گونه‌ای در کلاس‌های دانشگاه برگزار کرد به گونه‌ای که هیچ دو کلاسی با هم تداخل نداشته باشند؟**
**نشان دهید این مسئله NP-Complete است (می‌توانید از NP-Complete بودن مسئله‌ی k-coloring استفاده کنید). (۲۰ نمره)**

**نکته:** سوال بالا در زمان چندجمله‌ای و با استفاده از الگوریتم حریصانه قابل حل است. الگوریتم و اثبات این موضوع را می‌توانید در این لینک مشاهده کنید. به کسانی که ثابت کرده‌اند این مسأله در زمان چندجمله‌ای قابل حل است نمره‌ی کامل تعلق می‌گیرد. همچنین، به کسانی که مسأله‌ی k-coloring را به این مسأله کاهش داده‌اند نیز نمره‌ی کامل تعلق می‌گیرد. حال، مسئله را با فرض زیر در نظر میگیریم و ثابت می‌کنیم در دسته‌ی NP-Complete قرار دارد:
**در هر کلاس نباید بیشتر از m درس برگزار شود.**

پاسخ

ابتدا با ارائه دادن یک verifier ثابت می‌کنیم این مسئله NP است. با داشتن یک زمان بندی درست C برای n درس با بازه‌ی زمان‌بندی S و تعداد کلاس k، بررسی می‌کنیم که آیا تمامی کلاس‌های برگزار شده در یک کلاس تداخل دارند و آیا در هر کلاس کمتر یا مساوی m درس برگزار می‌شود. این بررسی در زمان $O(n^2)$ قابل انجام است زیرا کافیست بازه‌ی زمانی هر درس مربوط به یک کلاس با تمامی بازه‌های زمانی موجود در آن کلاس مقایسه شوند. از آنجایی که k کلاس داریم، این عملیات حداکثر k بار انجام خواهد شد پس یک verifier در زمان $O(kn^2)$ ارائه شد پس این مسئله NP است.

حال ثابت می‌کنیم این مسئله یک مسئله‌ی NP-Hard است. به این منظور، از مسئله‌ی k-coloring که یک مسئله‌ی NP-Complete است استفاده می‌کنیم و آن را به مسئله‌ی برنامه‌ریزی کلاس‌ها کاهش می‌دهیم. به این منظور، فرض می‌کنیم یک گراف ورودی شامل تعدادی رأس و یال و یک عدد k داریم. حال، می‌توان هر درس را به عنوان یک رأس، هر رنگ را به عنوان یک کلاس و هر یال را به عنوان وجود تداخل بین بازه‌های زمانی ۲ درس در نظر گرفت. این در واقع مسئله‌ی k-coloring است که در آن گراف خاصی با k رنگ مختلف، رنگ می‌شود. در این مسئله‌ی خاص، ما باید کمتر یا برابر با m رأس در هر رنگ داشته باشیم. به این منظور می‌توان با استفاده از قوانین استاندارد رنگ آمیزی به این امر دست پیدا کرد و به محض اینکه تعداد کلاس‌ها از m گذشت، رنگ را به رنگی جدید تغییر داد.

حال اگر یک k-coloring وجود داشته باشد، یعنی می‌توان با استفاده از k رنگ تمامی رأس‌ها را رنگ کرد یا به عبارتی دیگر، تمامی درس‌ها را برنامه‌ریزی کرد. پس می‌توان یک schedule معرفی کرد، پس $(S, k, n, m) \in Scheduling$.

اگر یک k-coloring وجود نداشته باشد، یعنی نمی‌توان با حداقل k رنگ تمامی رأس‌ها را رنگ کرد. پس نمی‌توان یک schedule معرفی کرد، پس $(S, k, n, m) \notin Scheduling$.

پس این مسئله یک مسئله‌ی NP-Complete است.