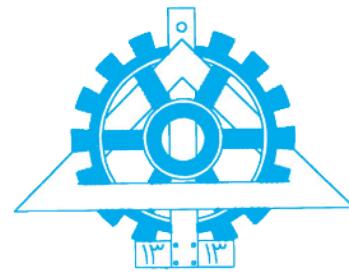




به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۱

دستیار آموزشی این مجموعه: معین کرمی

moein2000n@gmail.com

۱. زبان‌های توصیف شده را به صورت ریاضی نمایش دهید: (15)

$$\Sigma = \{a, b, \dots, z\}$$

a. زبان شامل رشته‌هایی که حرف اول و آخر آن‌ها یکی باشند.

$$L = \{w | \exists w', \lambda(w = \lambda.w'.\lambda \wedge w' \in \Sigma^* \wedge \lambda \in \Sigma)\}$$

b. زبان شامل رشته‌هایی که شامل ballas نباشد.

$$L = \{w | \nexists w_1, w_2 (w_1 \in \Sigma^* \wedge w_2 \in \Sigma^* \wedge w = w_1.ballas.w_2)\}$$

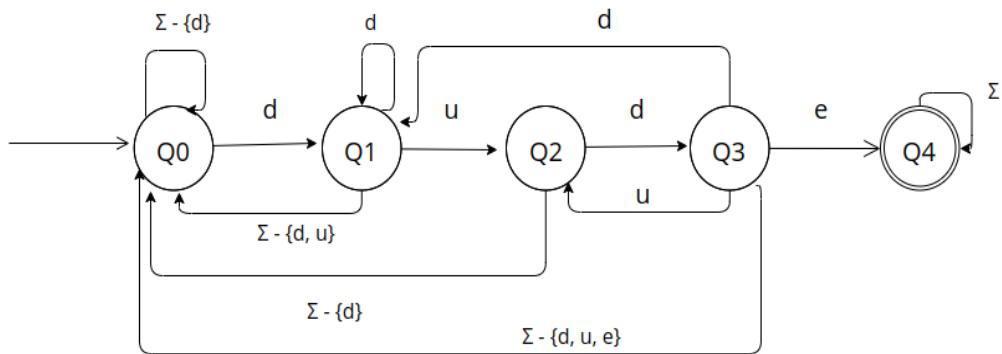
c. زبانی که هیچ یک از رشته‌های آن پسوند (suffix) رشته‌ی دیگری نباشد.

$$L = \{w | \nexists w_1, w_2 (w = w_1.w_2 \wedge w_2 \in L \wedge |w_1| > 0)\}$$

2. برای زبان‌های زیر DFA رسم کنید. (20)

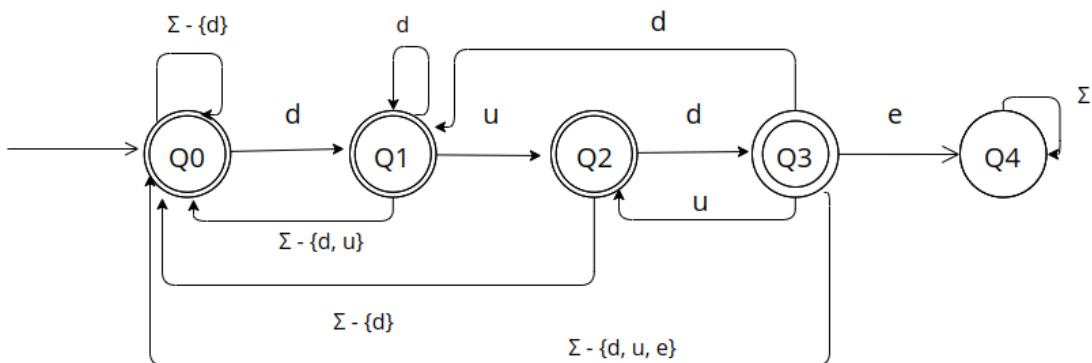
$$\Sigma = \{a, b, \dots, z\}$$

a. زبان شامل رشته‌هایی که شامل **dude** هستند. (با حداقل ۵ استیت)

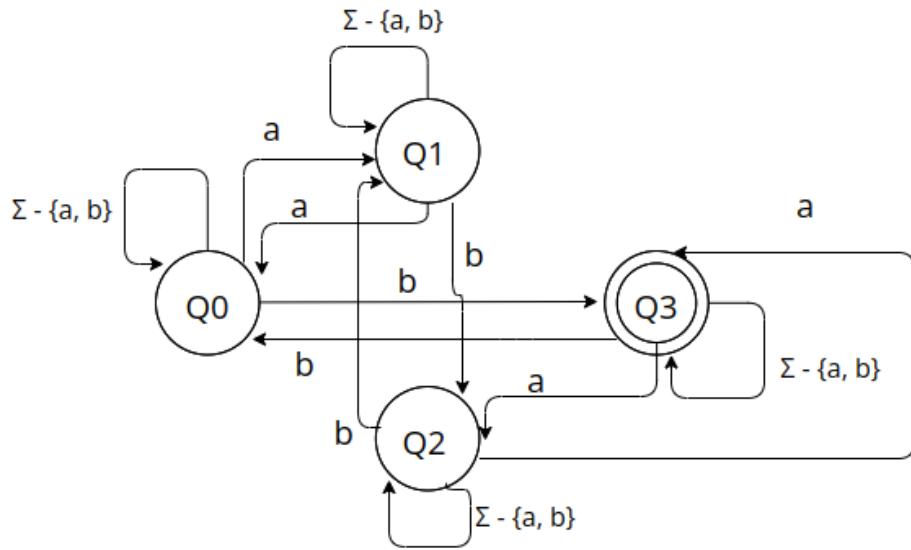


b. زبان شامل رشته‌هایی که شامل **dude** نیستند. (با حداقل ۵ استیت)

اگر کمی دقت کنید این زبان دقیقاً مکمل زبان قبلی است، پس با عوض کردن حالت استیت‌ها acceptable بودن یا نبودن) می‌توان به DFA این زبان رسید.



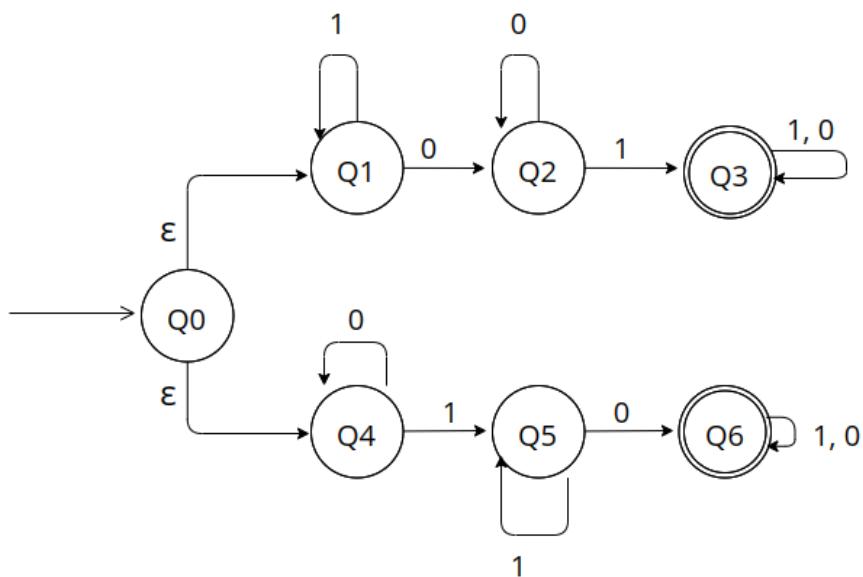
c. زبان شامل رشته‌هایی که شامل تعداد زوجی a و تعداد فردی b هستند. (با حداکثر ۴ استیت)



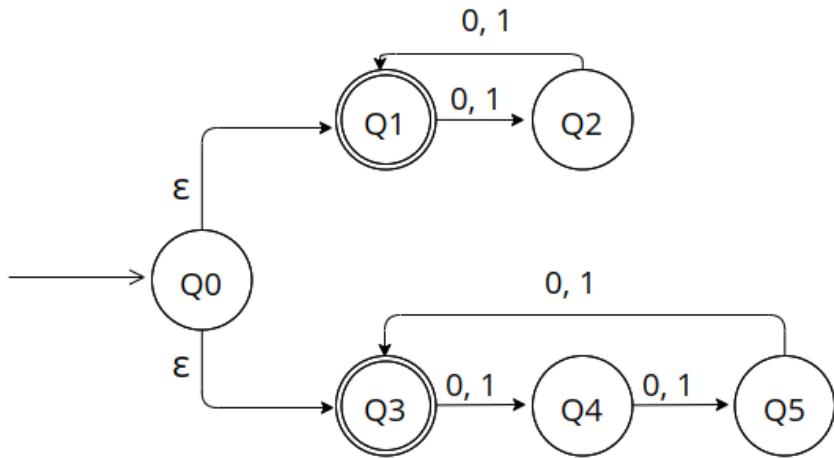
3. برای زبان‌های زیر NFA رسم کنید. (20)

$$\Sigma = \{0, 1\}$$

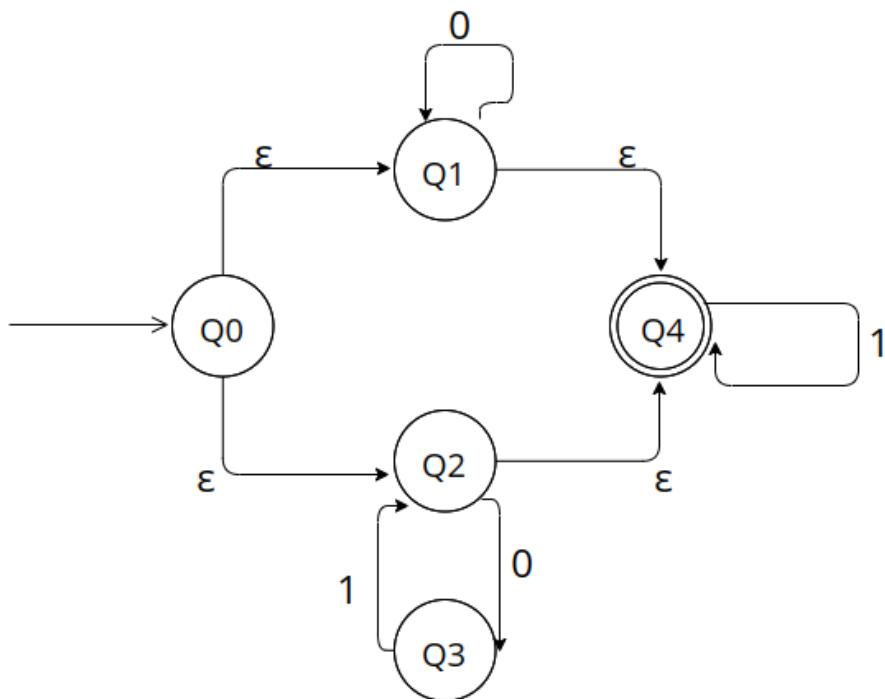
a. زبان شامل رشته‌هایی که شامل 01 و یا 10 باشند.



b. زبان هایی که شامل رشته هایی با طول مضرب ۲ یا ۳ باشند.



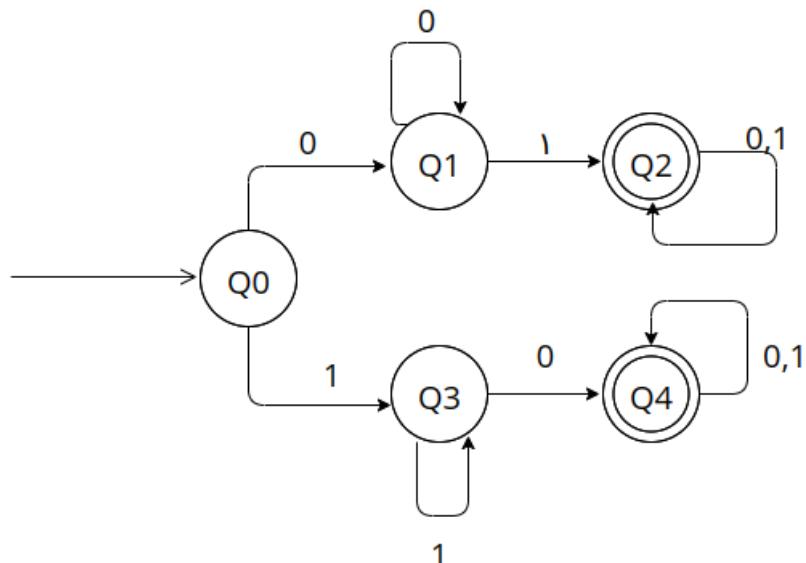
c. زبان هایی که بتوان آن ها را به دو زیر رشته متوالی تقسیم کرد که رشته اول یا تهی باشد، یا فقط شامل ۰ باشد و یا شامل تعدادی ۱ باشد و رشته‌ی دوم نیز یا تهی باشد و یا فقط شامل ۱ باشد. (با حداقل ۵ استیت)



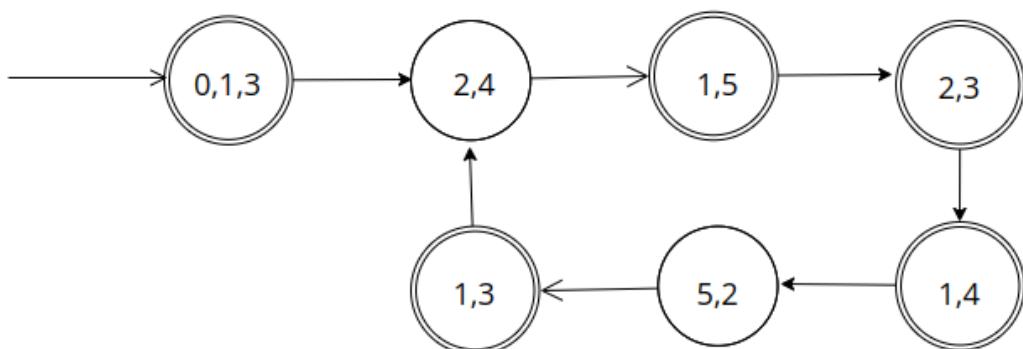
4. NFA های سوال قبل را به DFA تبدیل کنید. (20)

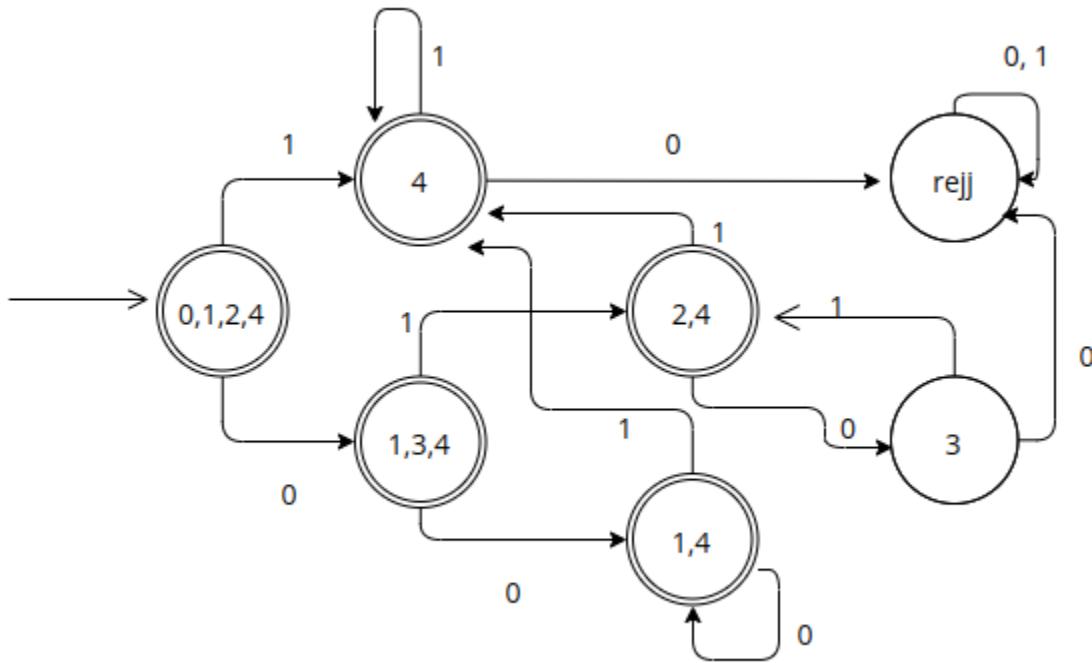
توجه کنید که حتما نیاز نیست برای این کار از الگوریتم تبدیل DFA به NFA استفاده کنیم، از همین رو برای زبان اول که از بقیه ساده تر است و میتوانستیم NFA آن را بسیار ساده تر بکشیم، بدون در نظر گرفتن DFA آن یک DFA میکشیم و برای دو زبان بعدی NFA را به DFA تبدیل میکنیم. دقت کنید اگر NFA مورد اول را به DFA تبدیل میکردیم به DFA بسیار پیچیده تری میرسیدیم.

_1



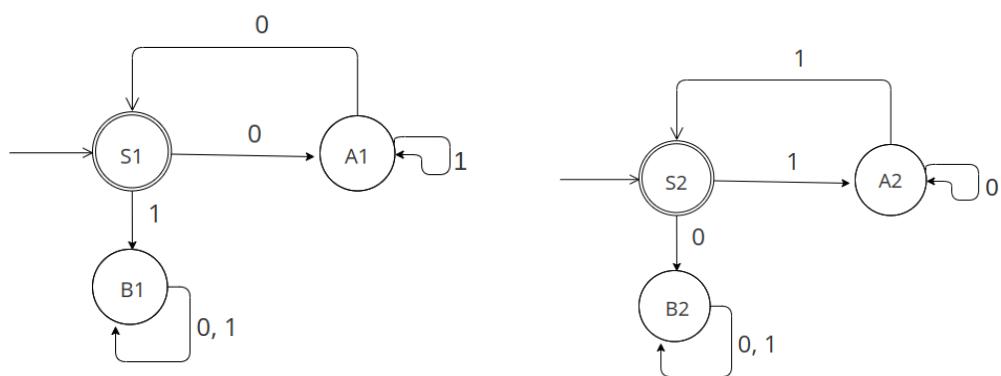
_2



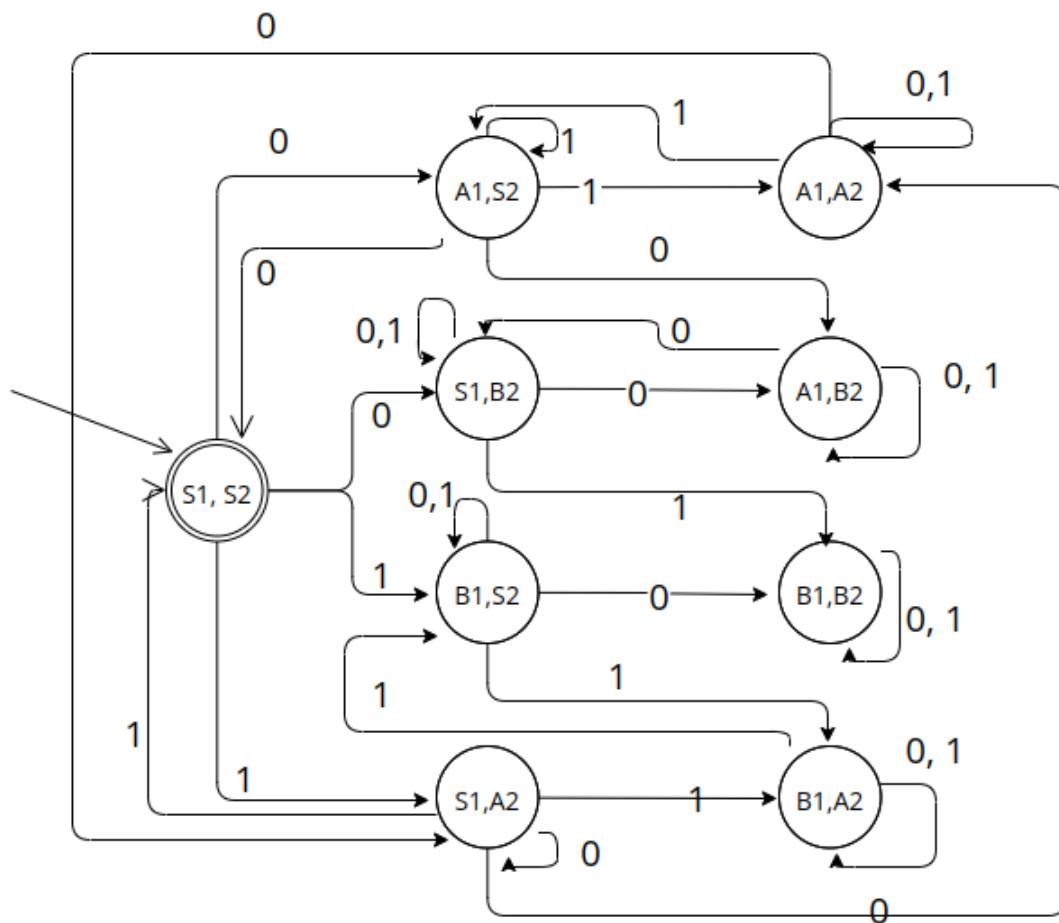


5. عملیات در هم ریختگی برای دو زبان منظم به این صورت تعریف می‌شود: دو زبان منظم A و B را در نظر بگیرید، زبان L که حاصل در هم ریختگی دو زبان A و B است شامل تمام رشته‌هایی است که بتوان آن رشته‌ها را به دو زیر دنباله افزای کرد به طوری که یکی از این زیر دنباله‌های (که خود یک رشته است) متعلق به زبان A و دیگری متعلق به زبان B باشد.

حال دو DFA زیر را در نظر بگیرید، NFA‌ای رسم کنید که نشان دهندهی زبان حاصل از در هم ریختگی زبان این دو DFA باشند. لطفا راه حل خود را نیز توضیح دهید. (حداکثر ۹ استیت)

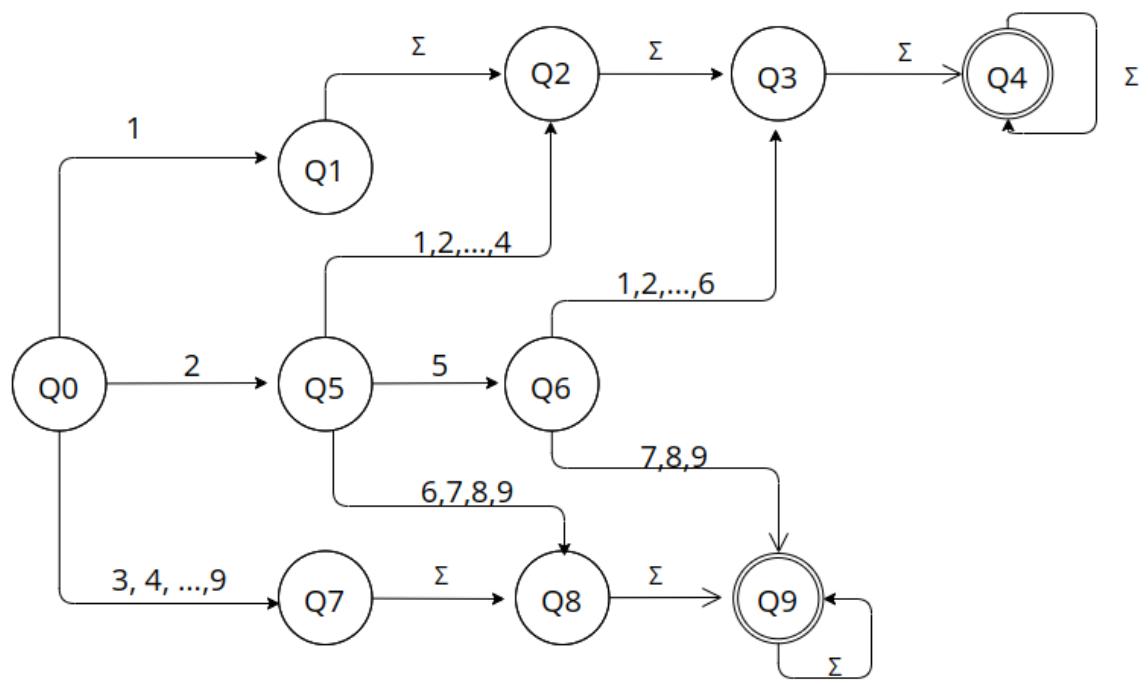


برای رسم NFA خواسته شده کافی است یک NFA با ۹ استیت در نظر بگیریم که هر استیت آن متناظر یک زوج مرتب از استیت‌های دو DFA بالاست، استیت ابتدایی برابر استیت متناظر با $S1, S2$ است و accepting استیت نیز معادل ترکیب همین دو استیت است. حال برای رسم transition‌ها کافی است بینیم با دیدن هر کاراکتر از هر زوج مرتب به کدام زوج‌ها می‌توان رفت و ترانزیشن‌ها را بر همین اساس رسم کرد.



6. برای زبانی که شامل اعداد بزرگتر از 256 باشند، یک DFA رسم کنید. (10)

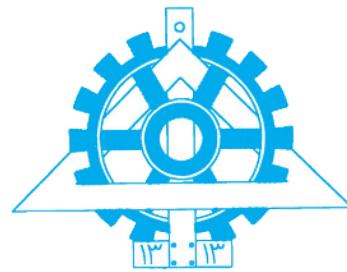
$$\Sigma = \{1, 2, \dots, 9\}$$





به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۱

دستیار آموزشی این مجموعه: معین کرمی

moein2000n@gmail.com

تاریخ تحويل: ۲۳:۵۹ ۱۴۰۱/۱۲/۹

۱. زبان‌های توصیف شده را به صورت ریاضی نمایش دهید: (۱۵)

$$\Sigma = \{a, b, \dots, z\}$$

a. زبان شامل رشته‌هایی که حرف اول و آخر آن‌ها یکی باشند.

b. زبان شامل رشته‌هایی که شامل ballas نباشد.

c. زبانی که هیچ یک از رشته‌های آن پسوند (suffix) رشته‌ی دیگری نباشد.

۲. برای زبان‌های زیر DFA رسم کنید. (۲۰)

$$\Sigma = \{a, b, \dots, z\}$$

a. زبان شامل رشته‌هایی که شامل dude هستند. (حداکثر ۵ استیت)

b. زبان شامل رشته‌هایی که شامل dude نیستند. (حداکثر ۵ استیت)

c. زبان شامل رشته‌هایی که شامل تعداد زوجی a و تعداد فردی b هستند. (حداکثر ۴ استیت)

۳. برای زبان‌های زیر NFA رسم کنید. (۲۰)

$$\Sigma = \{0, 1\}$$

a. زبان شامل رشته‌هایی که شامل ۰۱ و ۱۰ باشند.

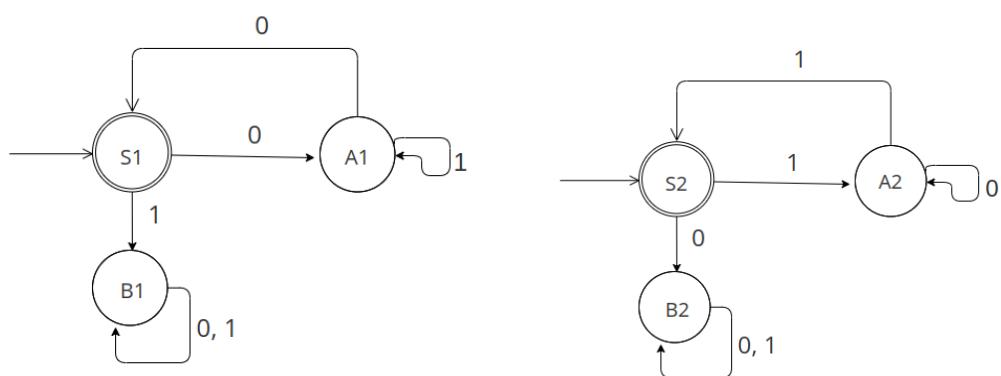
b. زبان‌هایی که شامل رشته‌هایی با طول مضرب ۲ یا ۳ باشند.

c. زبان‌هایی که بتوان آن‌ها را به دو زیر رشته متوالی تقسیم کرد که رشته اول یا تهی باشد، یا فقط شامل

۰ باشد و یا شامل تعدادی ۰۱ باشد و رشته‌ی دوم نیز یا تهی باشد و یا فقط شامل ۱ باشد. (با حداکثر ۵

استیت)

4. NFA های سوال قبل را به DFA تبدیل کنید. (20)
5. عملیات در هم ریختگی برای دو زبان منظم به این صورت تعریف می‌شود: دو زبان منظم A و B را در نظر بگیرید، زبان L که حاصل در هم ریختگی دو زبان A و B است شامل تمام رشته‌هایی است که بتوان آن رشته‌ها را به دو زیر دنباله افزای کرد به طوری که یکی از این زیر دنباله‌های (که خود یک رشته است) متعلق به زبان A و دیگری متعلق به زبان B باشد.
- حال دو DFA زیر را در نظر بگیرید، NFA ای رسم کنید که نشان دهنده زبان حاصل از در هم ریختگی زبان A و دو NFA باشد. لطفا راه حل خود را نیز توضیح دهید. (حداکثر ۹ استیت) (25)



6. برای زبانی که شامل اعداد بزرگتر از 256 باشد، یک DFA رسم کنید. (10)
- $$\Sigma = \{1, 2, \dots, 9\}$$

به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۲

پاسخ تمرین شماره ۲

دستیار آموزشی این مجموعه: پریا خوشتاب

paria.khoshtab2019@gmail.com



تاریخ تحولی: ۱۴۰۱/۱۲/۱۶

(۱) برای هر یک از زبان‌های زیر عبارت منظم بنویسید. (۳۰ نمره)

(الف) رشته‌هایی که تعداد a ها در آن‌ها فرد باشد. ($\Sigma = \{a, b, c\}$)

$$(b + c)^*a(a(b + c)^*a + (b + c))^*$$

(ب) اعداد باینری که مقدار آن‌ها در مبنای ده، زوج و بیشتر یا مساوی ۸ باشد. ($\Sigma = \{0, 1\}$)

$$0^*1(0 + 1)^*(0 + 1)(0 + 1)0$$

(ج) رشته‌هایی که شامل زیررشته bc نمی‌باشند. ($\Sigma = \{a, b, c\}$)

$$c^*(b + ac^*)^*$$

(د) زبانی که شامل تمامی رشته‌ها به جز aaa باشد. ($\Sigma = \{a, b\}$)

$$\epsilon + a + aa + (b + ab + aab + aaa(a + b))(a + b)^*$$

(ه) رشته‌هایی که حداقل شامل دو زیررشته aa باشند. ($\Sigma = \{a, b\}$)

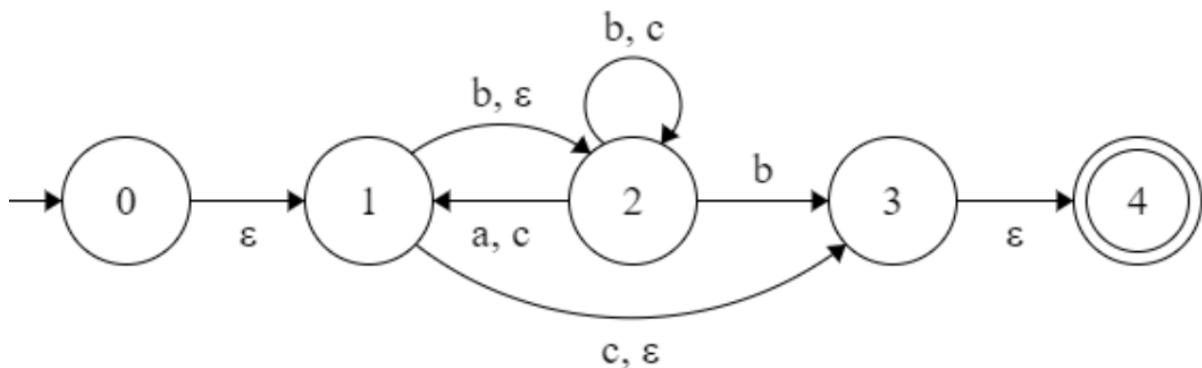
$$(a + b)^*aa(a + b)^*aa(a + b)^* + (a + b)^*aaa(a + b)^*$$

(و) (امتیازی) رشته‌هایی که شامل تعداد زوجی زیررشته ۰۰۰ می‌باشند. ($\Sigma = \{0, 1\}$)

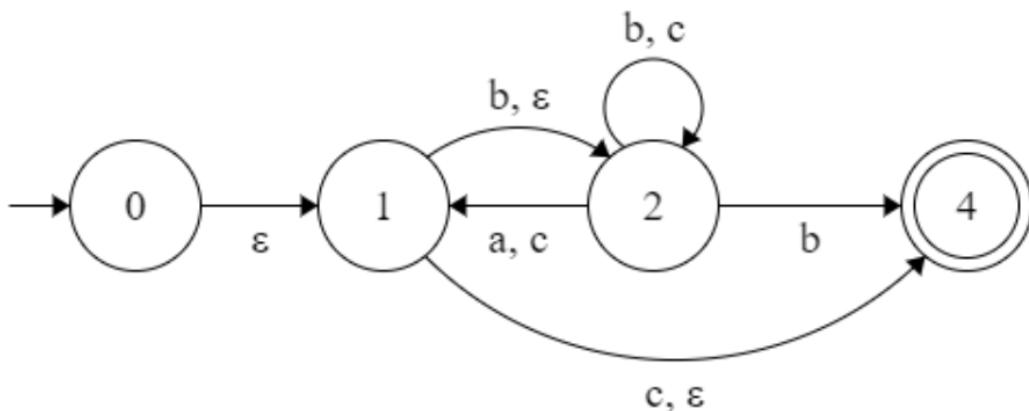
$$(((0 + (00)^*)1)^*000(00)^*1((0+(00)^*1)^*000(00)^*1)^*((0+(00)^*)1)^*(0+(00)^*)$$

(2) عبارت منظم متناظر با هر یک از NFA های زیر را بنویسید و مراحل تبدیل و حذف هر state را نیز رسم کنید. (30 نمره)

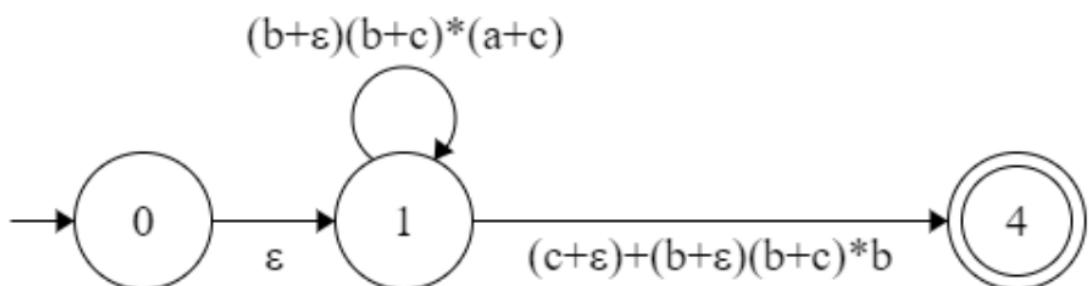
الف) ابتدا تبدیل به GNFA را انجام می دهیم:



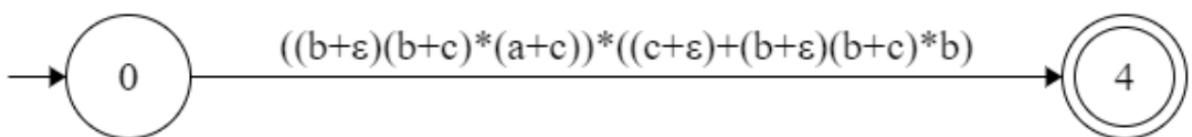
سپس استیت 3 را حذف می کنیم:



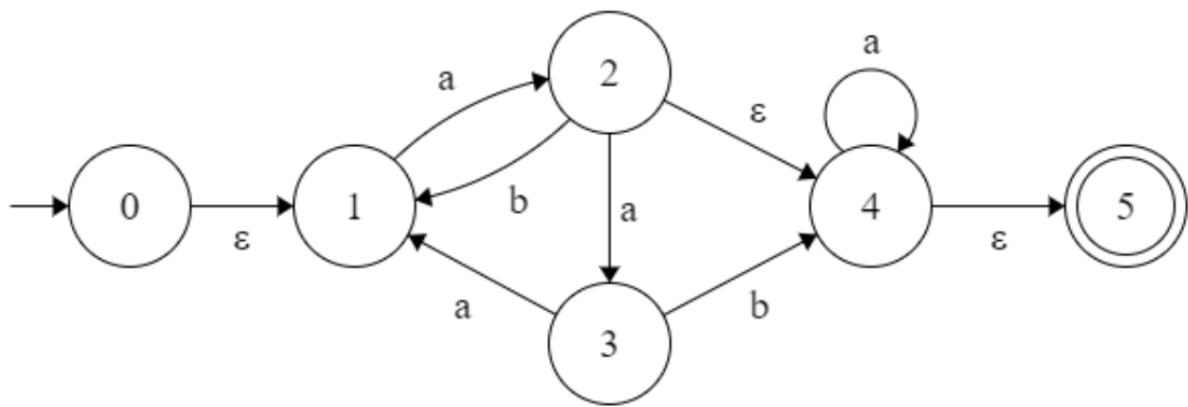
سپس استیت 2 را حذف می کنیم:



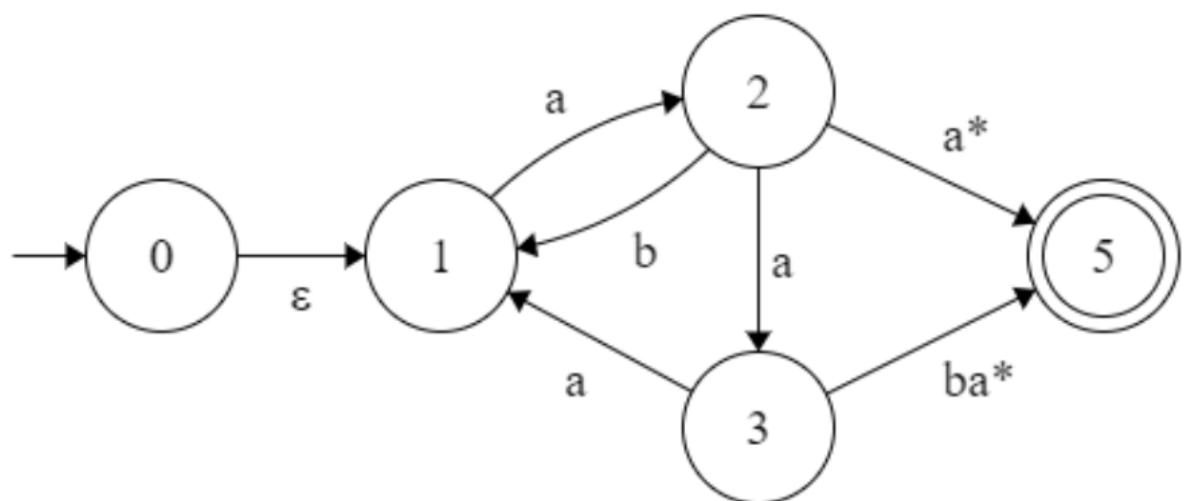
سپس استیت 1 را حذف می کنیم:



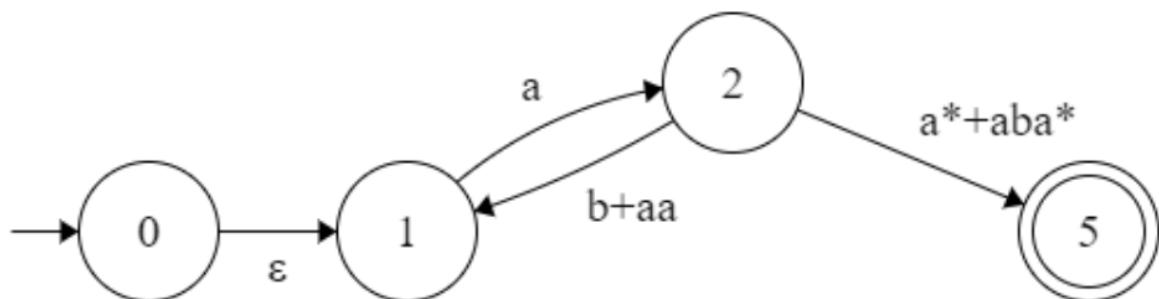
ب) ابتدا تبدیل به GNFA را انجام می‌دهیم:



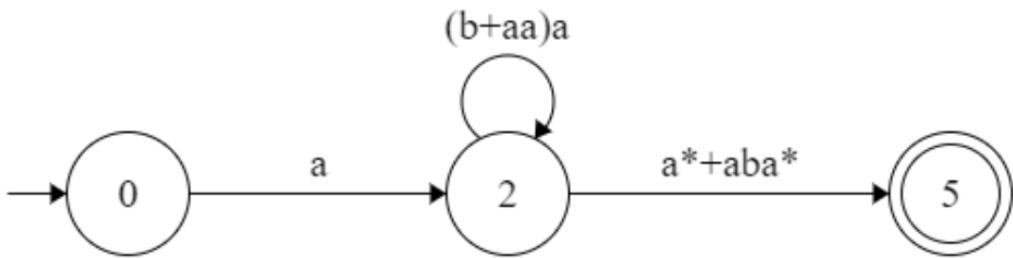
سپس استیت 4 را حذف می‌کنیم:



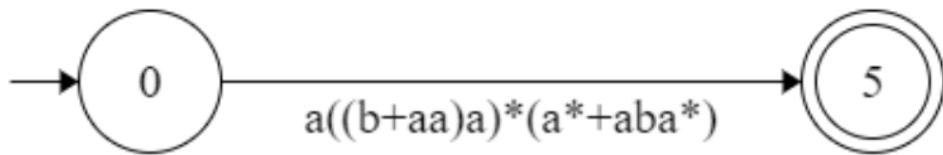
سپس استیت 3 را حذف می‌کنیم:



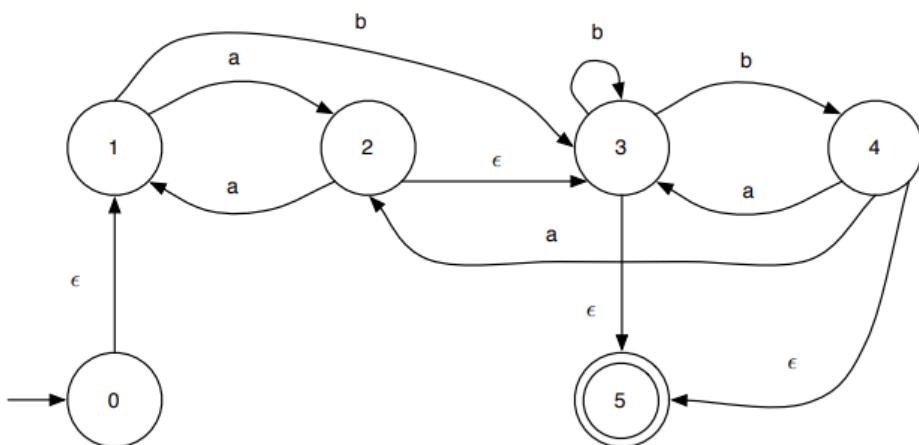
سپس استیت 1 را حذف می‌کنیم:



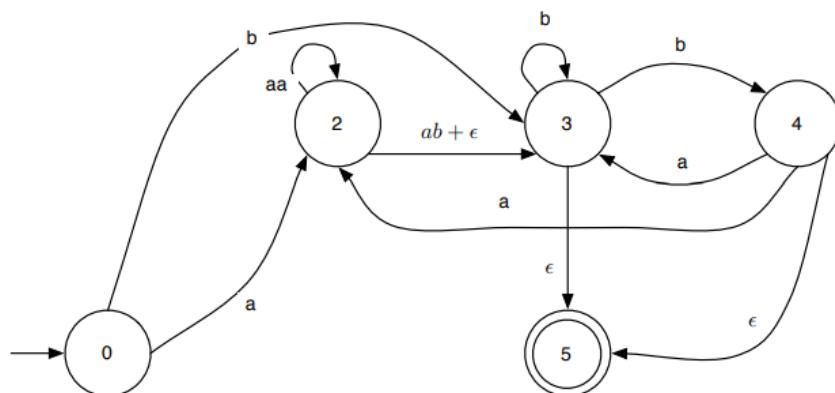
سپس استیت 2 را حذف می‌کنیم:



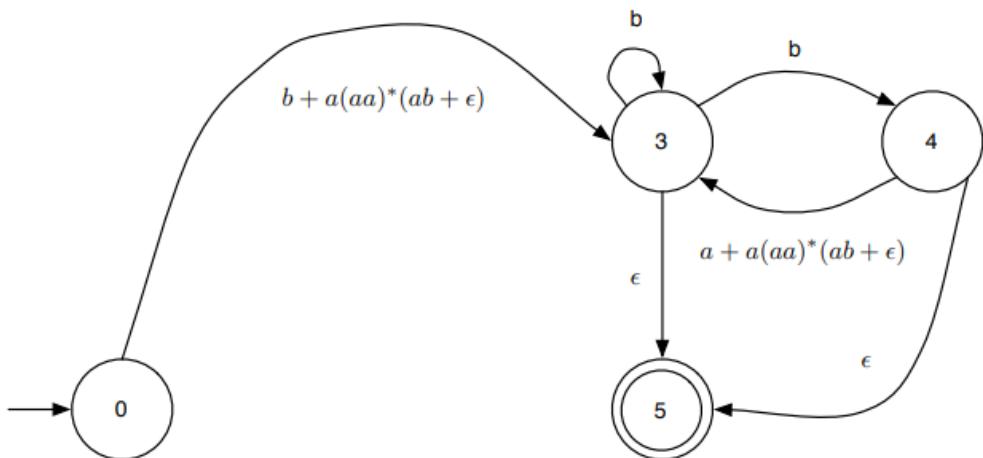
ج) ابتدا تبدیل به GNFA را انجام می‌دهیم:



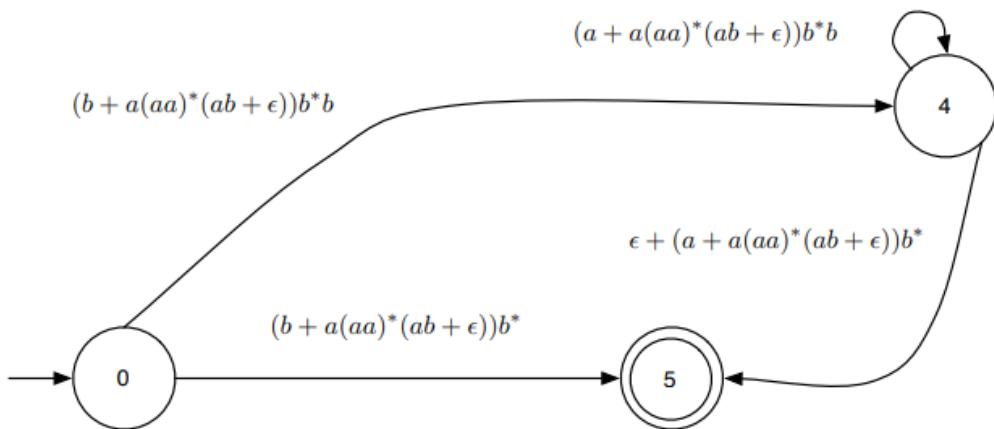
سپس استیت 1 را حذف می‌کنیم:



سپس استیت 2 را حذف می‌کنیم:



سپس استیت 3 را حذف می‌کنیم:



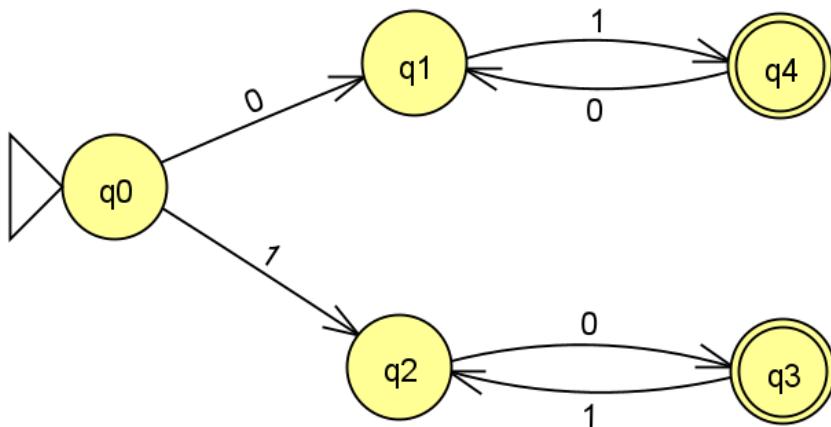
سپس استیت 4 را حذف می‌کنیم:

$$(b + a(aa)^*(ab + \epsilon))b^* + ((b + a(aa)^*(ab + \epsilon))b^*)((a + a(aa)^*(ab + \epsilon))b^*)^*(\epsilon + (a + a(aa)^*(ab + \epsilon))b^*)$$

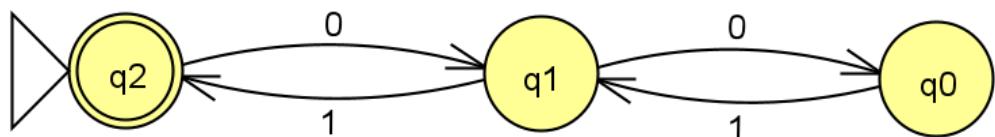


(3) برای عبارات منظم زیر DFA رسم کنید. (20 نمره) (استیت Trap به منظور سادهسازی رسم نشده است)

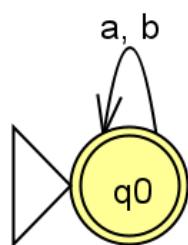
الف) $0(10)^*1 + 1(01)^*0$



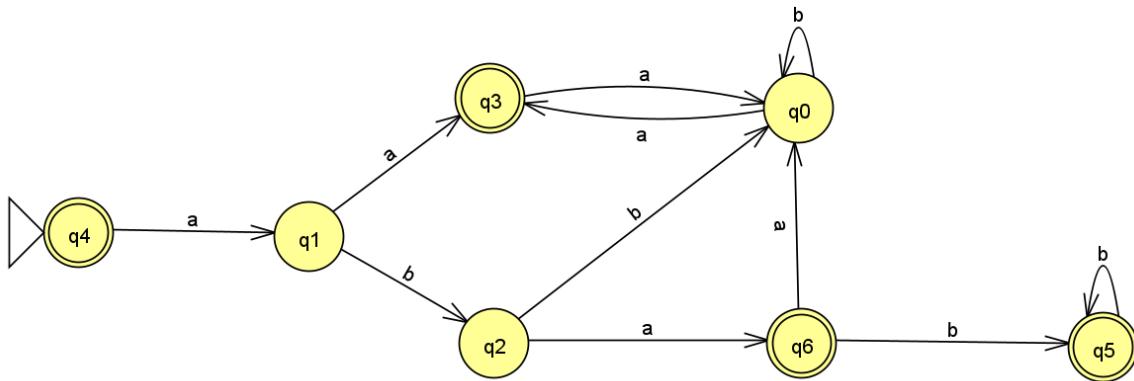
$(0(01 + 10)^*1)^*$ (ب)



$((a^*b)^*(b^*a)^*)^*$ (ج)



د) (امتیازی) $abab^* + (ab^*a)^*$



4) درستی یا نادرستی موارد زیر را مشخص کنید. (در صورت نادرست بودن مثال نقض و در صورت درستی اثبات ارائه دهید.) (20 نمره)

الف) اگر $L_1 \cap L_2$ زبان‌های منظم باشند، زبان L_1 نیز منظم است.
نادرست؛ مثال نقض:

$$L_1 = \{a^{2^n} : n \geq 0\}, L_2 = \emptyset, L_1 \cap L_2 = \emptyset$$

ب) اگر L^* منظم باشد، زبان L نیز منظم است.
نادرست؛ مثال نقض:
زبان $\{\epsilon\}$ نامنظم است ولی $L^* = \{a^{2^n} : n \geq 0\}$ به وضوح منظم است.

ج) اگر L_1 و L_2 زبان‌های منظم باشند، $L_1 \setminus L_2$ نیز منظم است. (عملگر \ نشان‌دهنده تقاضل مجموعه‌ای است)
درست؛ اثبات:

می‌دانیم زبان‌های منظم نسبت به اعمال اشتراک، اجتماع و متام بسته‌اند و چون $\overline{L_2} = L_1 \cap \overline{L_2}$ می‌دانیم نسبت به تقاضل مجموعه‌ای نیز بسته‌اند.
بنابراین نسبت به تقاضل مجموعه‌ای نیز بسته‌اند.

د) اگر $L \setminus \{\epsilon\}$ منظم باشد، زبان L نیز منظم است. (عملگر \ نشان‌دهنده تقاضل مجموعه‌ای است)
درست؛ اثبات:
طبق فرض می‌دانیم $L \setminus \{\epsilon\}$ منظم است. همچنین می‌دانیم $\{\epsilon\}$ نیز منظم است. بنابراین با توجه به بسته بودن زبان‌های منظم نسبت به عمل اجتماع، $L = L \setminus \{\epsilon\} \cup \{\epsilon\}$ نیز منظم است.

ه) اگر L_1 زبان منظم و $L_2 \subseteq L_1$ باشد، زبان L_2 نیز منظم است.
نادرست؛ مثال نقض:

$$L_1 = a^*, L_2 = \{a^{2^n} : n \geq 0\}, L_2 \subseteq L_1$$

(5) فرض کنید A یک زبان منظم باشد و B هر زبانی باشد (ازوماً منظم نیست). ثابت کنید زبان L منظم می‌باشد.
 (10 نمره)

$$L = \{w \mid wx \in A \text{ به طوریکه } x \in B\}$$

یک زبان منظم می‌باشد، بنابراین یک DFA به نام M وجود دارد که زبان A را می‌پذیرد.
 فرض می‌کنیم M داریم: $M = (Q, \Sigma, \delta, q_0, F)$

برای این که ثابت کنیم L یک زبان منظم است باید یک DFA بسازیم که زبان L را پذیرد. این DFA به نام M' را به صورت زیر می‌سازیم:

- $M' = (Q', \Sigma', \delta', q'_0, F')$
- $Q' = Q$
- $\Sigma' = \Sigma$
- $q'_0 = q_0$
- $F' = \{q \in Q \mid \exists x \in B : \hat{\delta}(q, x) \in F\}$



تاریخ تحويل: ۱۴۰۱/۱۲/۱۶

(۱) برای هر یک از زبان‌های زیر عبارت منظم بنویسید. (30 نمره)

(الف) رشته‌هایی که تعداد a ‌ها در آن‌ها فرد باشد. ($\Sigma = \{a, b, c\}$)

(ب) اعداد باینری که مقدار آن‌ها در مبنای ده، زوج و بیشتر یا مساوی 8 باشد. ($\Sigma = \{0, 1\}$)

(ج) رشته‌هایی که شامل زیررشته bc نمی‌باشند. ($\Sigma = \{a, b, c\}$)

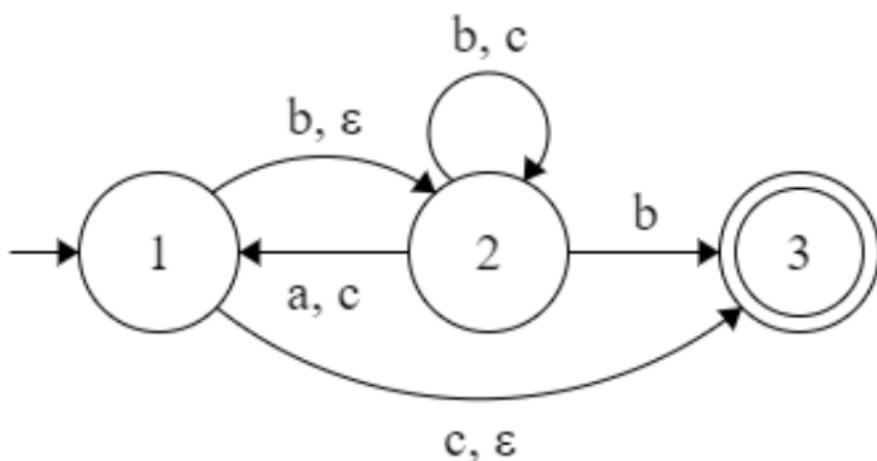
(د) زبانی که شامل تمامی رشته‌ها به جز aaa باشد. ($\Sigma = \{a, b\}$)

(ه) رشته‌هایی که حداقل شامل دو زیررشته aa باشند. ($\Sigma = \{a, b\}$)

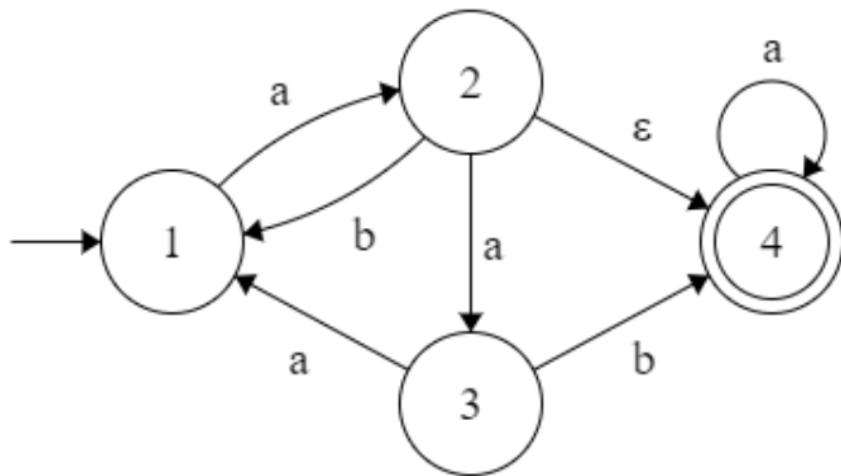
(و) (امتیازی) رشته‌هایی که شامل تعداد زوجی زیررشته 000 می‌باشند. به عنوان مثال 0001000 و 0000 عضو این زبان هستند ولی 00000 عضو این زبان نیست. ($\Sigma = \{0, 1\}$)

(۲) عبارت منظم متاظر با هر یک از NFA‌های زیر را بنویسید و مراحل تبدیل و حذف هر state را نیز رسم کنید. (30 نمره)

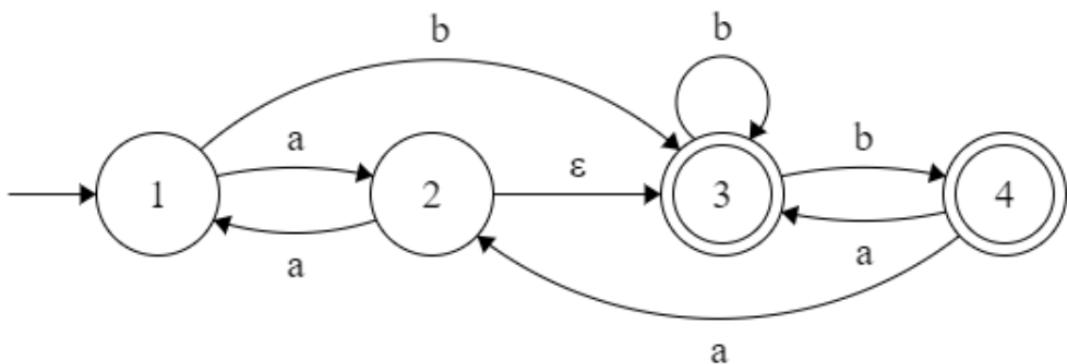
(الف)



(ب)



(ج)



(3) برای عبارات منظم زیر DFA رسم کنید. (20 نمره)

(الف) $0(10)^*0 + 1(01)^*1$

(ب) $(0(01 + 10)^*1)^*$

(ج) $((a^*b)^*(b^*a)^*)^*$

(د) **امتیازی** $abab^* + (ab^*a)^*$

(4) درستی یا نادرستی موارد زیر را مشخص کنید. (در صورت نادرست بودن مثال نقض و در صورت درستی اثبات ارائه دهید). (20 نمره)

(الف) گر $L_1 \cap L_2$ و L_2 زبان‌های منظم باشند، زبان L_1 نیز منظم است.

(ب) اگر L^* زبان منظم باشد، زبان L نیز منظم است.

ج) اگر L_1 و L_2 زبان‌های منظم باشند، $L_1 \setminus L_2$ نیز منظم است. (عملگر \setminus نشان‌دهنده تقاضل مجموعه‌ای است)

د) اگر $\{L\} \cup \{\epsilon\}$ منظم باشد، زبان L نیز منظم است. (عملگر \cup نشان‌دهنده تقاضل مجموعه‌ای است)

ه) اگر L_1 زبان منظم و $L_2 \subseteq L_1$ باشد، زبان L_2 نیز منظم است.

(5) فرض کنید A یک زبان منظم باشد و B هر زبانی باشد (ازوماً منظم نیست). ثابت کنید زبان L منظم می‌باشد.
(10 نمره)

$$L = \{w \mid wx \in A \text{ به طوریکه } x \in B\}$$



به نام خدا

۱۴۰ نظریه زبان ها و ماشین ها - بهار

پاسخ تمرین شماره سه

:ایمیل دستیار آموزشی این مجموعه



f24moh@gmail.com

- در مواردی که خواسته شده نامنظم بودن یا نبودن زبانها را اثبات کنید، نامنظم بودن هیچ زبانی را به صورت پیش فرض در نظر نگیرید.
- یکی از اهداف این تمرین یادگیری "الم تزريق" میباشد، درصورتی که از روش دیگری برای نشان دادن نامنظم بودن زبانها استفاده کنید نمره‌ای به حل شما تعلق نمیگیرد.
- برای نشان دادن منظم بودن زبانها کافی است DFA آنرا رسم کنید و توجه کنید DFA به صورت کمینه شده تنها قابل قبول است.

(1) نامنظم بودن زبان‌های زیر را اثبات کنید.

a) $L_1 = \{0^i 1^j \mid \gcd(i, j) = 1\}$

1. devil: picks p

2. you: $w = 0^{p'} 1^{(p'-1)!}$,

P' اولین عدد اول بزرگتر از p میباشد.

3. devil: $w = xyz$, $|xy| \leq p$, $|y| \neq 0$
 $\Rightarrow y = 0^k$; $k \geq 1$

4. you: $i = 0$

$\Rightarrow w' = xy^0 z = 0^{(p'-k)} 1^{(p'-1)!}$

$\Rightarrow \gcd(p-k, (p-1)!) = p - k$; $k \geq 1$

$\Rightarrow w' \notin L_1$

$\Rightarrow L_1$ is not regular.

b) $L_2 = \{(ab)^n a^k : n > k, k \geq 0\}$

1. devil: picks p

2. you: $w = (ab)^{p+1} a^p$

3. devil: $w = xyz$, $|xy| \leq p$, $|y| \neq 0$

چون طول رشته xy بایستی از p کمتر باشد هر دو جز رشته ساخته شده از $(ab)^{p+1}$ خواهند بود

4. you:

برای y چندین حالت داریم:

- $y = a \dots a$ or a شروع میشود و با a تمام میشود

در این صورت با قرار دادن $0 = i$ رشته‌ی ایجاد شده شامل bb میشود که واضح است این رشته نمیتواند جز زبان باشد.

- $y = (ba)^m$ یا $y = (ab)^m$

در این صورت با قرار دادن $0 = i$ رشته‌ی ایجاد شده دارای تعداد بیشتری a در انتهای رشته نسبت به ab ها میباشد و دیگر جز زبان نیست.

- $y = b \dots b$ or b

در این صورت با قرار دادن $0 = i$ رشته‌ی ایجاد شده شامل aa میشود که واضح است این رشته نمیتواند جز زبان باشد. (چرا که یا دو aa بین دو bb داریم و یا در اخر رشته داریم که در حالت اول شکل کلی رشته جز زبان نیست و در حالت دوم تعداد a ها بیشتر از ab ها میشود)

با توجه به اینکه ما در هر حال استراتژی برد داریم، زبان نامنظم است

c) $L_3 = \{w | w \in \{a, b, c\}^* \text{ and } n_a(w) \leq n_b(w) \leq n_c(w)\}$

1. devil: picks p

2. you: $w = a^p b^p c^p$

3. devil: $w = xyz, |xy| \leq p, |y| \neq 0$
 $\Rightarrow y = a^k$

4. you: $i = 2$

$\Rightarrow \text{new string } w' = a^{k+p} b^p c^p, k \neq 0 \Rightarrow n_a(w') > n_b(w) \Rightarrow w' \notin L_3$

در نتیجه زبان مورد نظر نامنظم است.

منظم بودن یا نامنظم بودن زیان‌های زیر را مشخص کنید. (پاسخ خود را اثبات کنید). (2)

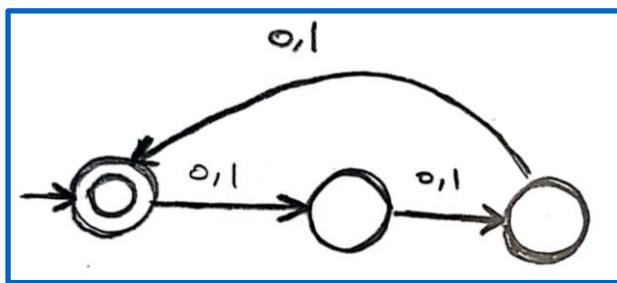
a) $L_1 = \{w_1 w_2 | w_1, w_2 \in \{0,1\}^* \text{ and } |w_1| = 2|w_2|\}$

منظم است.

درواقع زیان داده شده را میتوان به صورت زیر نوشت:

$L_1 = \{w | w \in \{0,1\}^* \text{ and } |w| = 3k, k \geq 0\}$

که میتوان DFA ازرا به صورت زیر رسم کرد:



b) $L_2 = \{a^n b^m c^k | n = m \text{ and } n \neq k\}$

نامنظم است.

1. devil: picks p

2. you: $w = a^p b^p c^{2p}$

3. devil: $w = xyz, |xy| \leq p, |y| \neq 0$

4. you: $i = 0 \Rightarrow xy^i z = a^{p-|y|} b^p c^{2p}$

باتوجه به اینکه $|y|$ صفر نمیباشد، رشته جدید نمیتوان جز زیان مورد نظر باشد، زیان نامنظم است

c) $L_3 = \{w | w \in \{a, b, c\}^*, |w| \text{ is prime number}\}$

نامنظم است.

1. devil: picks p

2. you: $w = a^q, q \text{ is prime number}, q > p$

3. devil: $w = xyz, |xy| \leq p, |y| \neq 0$

$\Rightarrow y = a^k; k \geq 1$

4. you: $i = q + 1$

$\Rightarrow w' = xy^i z = a^{q+qk} = a^{q(k+1)}$

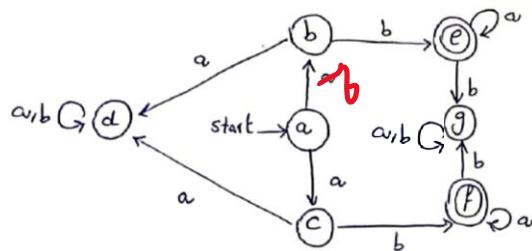
$\Rightarrow w' \notin L_3$

$\Rightarrow L_3 \text{ is not regular}$

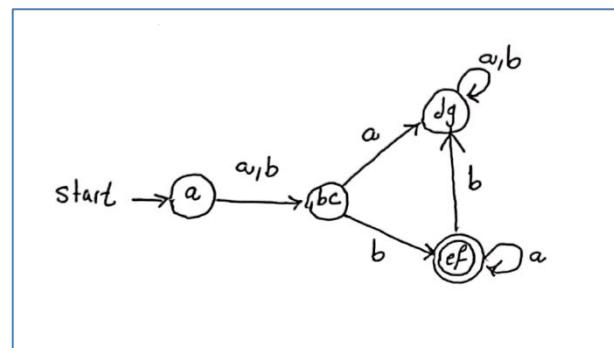
توجه شود هیچ یک از $k+1$ یا q نمیتوانند یک باشند، پس $(q)(k+1)$ اول نیست.

های داده شده را کمینه کنید.

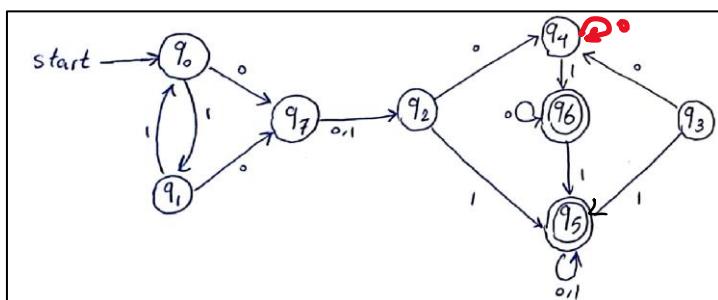
a) $\Sigma = \{a, b\}$



a						
	b					
	bc	c				
		d				
			e			
			ef	f		
		dg				g

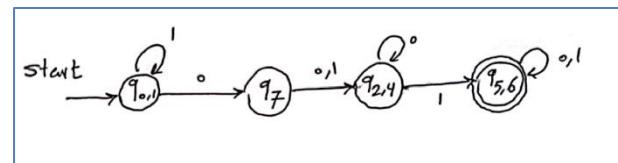


b) $\Sigma = \{0,1\}$

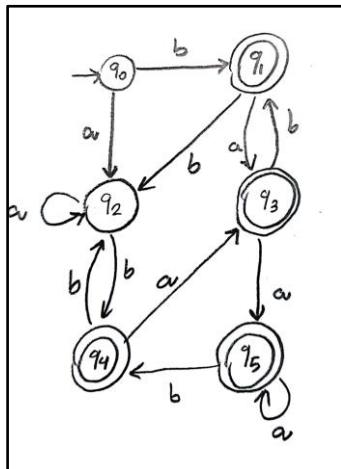


با توجه به اینکه مسیری از q_0 به q_3 (هیچ مسیری برای رسیدن به این استیت وجود ندارد)، این استیت قابل حذف از DFA میباشد، پس از حذف این استیت، استیت های برابر را پیدا میکنیم.

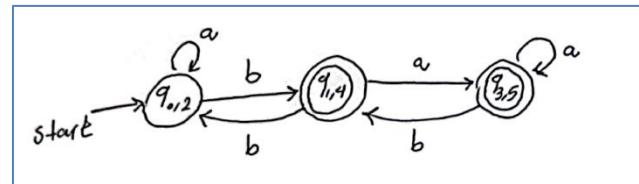
q0							
q0,1	q1						
		q2					
-	-	-	q3				
		q2,4	-	q4			
			-		q5		
			-		q5,6	q6	
			-				q7



c) $\Sigma = \{a, b\}$



q0							
	q1						
q0,2		q2					
			q3				
q1,4				q4			
		q3,5					q5



(4) یک دانشجو علاقمند به مباحث تئوری علوم کامپیوتر با استفاده از لم تزریق میخواهد اثبات کند که این زبان نامنظم است:

$$L = \{w_1 w_2 \mid w_i \in \{a, b\}^*, n_a(w_1) = n_b(w_2)\}$$

پاسخ این دانشجو به صورت زیر است:

1. حرفی مقدار $p \geq 1$ انتخاب میکند
2. من رشته $a^p b^p = w$ را انتخاب میکنم
3. حرفی رشته w را به صورت $x y z$ تقسیم بنده میکند و چون طول $x y$ کمتر p باید باشد، y حتماً به صورت a^p میباشد.
4. اگر من $0 = i$ قرار بدهم، تعداد a ها کمتر از تعداد b ها میشود و دیگر این رشته متعلق به زبان نمیباشد، پس زبان نامنظم است.

یا از نظر شما این استدلال صحیح است؟

اگر خیر اشکال کار کجاست؟

(اگر فکر میکنید زبان منظم است کافی است DFA مربوط به زبان را رسم کنید)

استدلال صحیح نیست؛

1. طول y میتواند متغیر باشد.

2. در مرحله 4 ام میتوانست رشته را به گونه ای بشکند که جز زبان باشد.

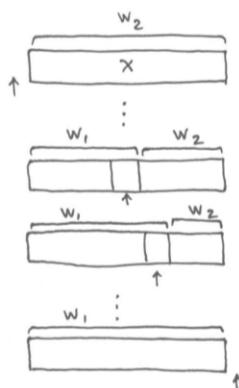
زبان منظم است.

اثبات:

$x = w_1 w_2 = \sum_{i=1}^n a_i b_i$ ی توان هر رشته دلخواه $x \in \sum_{i=1}^n a_i b_i$ صورت را داشت. این زبان منظم است.

اثبات. نرض کنیم برای رشته دلخواه $x = a_1 b_1 a_2 b_2 \dots a_n b_n$ باشد.

همگام خواهد x از $\sum_{i=1}^n a_i b_i$ براست مقادیر $n_a(w_1) - n_b(w_2)$ را در ترتیبی $\langle a_1, b_1, a_2, b_2, \dots, a_n, b_n \rangle$ میگیریم.

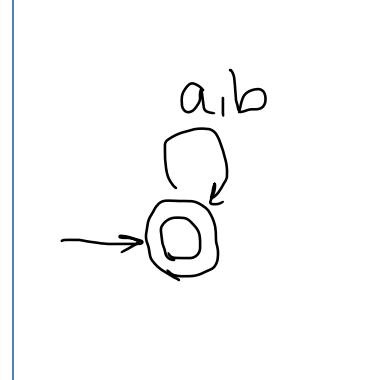


$$\begin{aligned} & n_a(w_1) - n_b(w_2) \\ & - B \\ & \vdots \\ & Y \\ & \vdots \\ & Y-1 \text{ یا } Y+1 \\ & \vdots \\ & A \end{aligned}$$

در هرین خواهد x از $\sum_{i=1}^n a_i b_i$ براست، مقدار $n_a(w_1) - n_b(w_2)$ از عدد متفق (-B).

بعدی مثبت (A) یا طور میوسته (اصناعه یا کم سدن یک راهد در مرحله) می رسد.

در این تغییرات یا تاها در جای مقدار آن صفری شود.



(5) زیان $L_{p(n)}$ به صورت زیر تعریف میشود، به این صورت که $p(n)$ یک چند جمله ای با ضرایب طبیعی باشد.

$$L_{p(n)} = \{w \mid w = 0^{p(n)}, n > 0, n \in N\}$$

ثابت کنید زیان $L_{p(n)}$ منظم است اگر و تنها اگر درجه چند جمله ای $p(n)$ برابر صفر یا یک باشد.

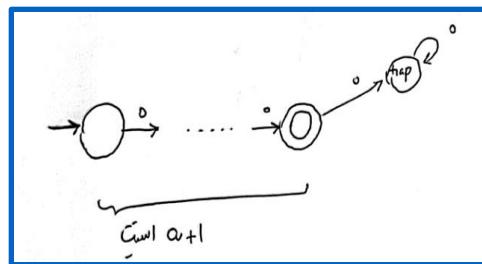
کافی است منظم بودن زیان مورد نظر را برای سه حالت که درجه چند جمله ای صفر - یک - یا بیشتر از یک باشد را بررسی کنیم.

اگر درجه چند جمله ای صفر باشد: ●

$$p(n) = a$$

$$L = \{0^a\}$$

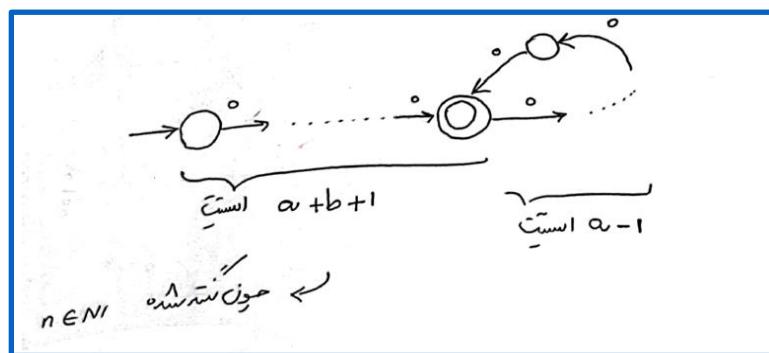
این زیان تنها یک عضو دارد و متناهی است و واضح است که منظم میباشد. (میتوان DFA مطابق زیر برای این زیان رسم کرد:)



اگر درجه چند جمله ای یک باشد: ●

$$p(n) = an + b$$

برای این زیان نیز میتوان DFA زیر را رسم کرد:



اگر درجه چند جمله ای بیشتر از یک باشد: ●

با استفاده از لم تزریق نشان میدهیم که که نمیتوانم منظم باشد:

1. devil: picks q

2. you: $w = 0^{p(q)}$

3. devil: $w = xyz$, $|xy| \leq p$, $|y| \neq 0$

$$4. \text{you: } w' = xy^i z, i = a$$

$$\Rightarrow |w'| = a|y| + |x| + |z| = a|y| + |xz| = an + b$$

با توجه به اینکه اندازه رشته $w' = an + b$ است، طول رشته جدید با افزایش a به صورت خطی افزایش پیدا میکند و مقدار آن روی خط با شبیه و عرض از مبدأ b قرار میگیرد.

همچنین میدانیم برای اینکه زیان مرود نظر منظم باشد، باید رشته جدید $xy^i z = w'$ عضو زیان L باشد، پس باید به ازای هر مقدار $i = a$ مقدار b برابر با روی منحنی $p(n)$ قرار بگیرد، درصورتی که که میدانیم رشد چند جمله ای ها با درجه بیشتر از یک مسلماً بیشتر از چند جمله ای با درجه یک میباشد و نمیتوان تمام مقادیر $an + b$ را روی یک چند جمله با درجه بیشتر از یک و با ضرایب ثابت منطبق کرد، پس این زیان نمیتواند منظم باشد.

(امتیازی) زیان (L) را به صورت زیر تعریف میکنیم:

$$L(K) = \{w_1 w_2 \mid w_i \in (0+1)^*, |w_2| = m, w_1 \text{ ends with } 0\}$$

اثبات کنید DFA کمینه حداقل 2^{m+1} استیت دارد.

این زیان رشته هایی را توصیف میکند که حرف $m+1$ ام از آخر آن برابر a است.

برهان خلف: فرض کنید DFA کمینه این زیان کمتر از 2^{m+1} استیت دارد.

چون در مجموع 2^{m+1} ترکیب مختلف برای $1+m$ حرف آخری که تاکنون خوانده شده وجود دارد. طبق اصل الانه کبوتری، استیقی وجود دارد که پس از خواندن دو دنباله $m+1$ حرف متفاوت به آن میرسیم.

این استیت را با q و این دو دنباله را با $k_1, k_2 \dots k_{m+1}$ نشان میدهیم. چون دو دنباله متمایز

هستند، حداقل در یک حرف تفاوت دارند. اولین نقطه تفاوت آنها را i مینامیم. به دلیل تقارن میتوان فرض کرد که

$$n_i = b, k_i = a$$

استیقی که با خواندن دنباله b^{i-1} با شروع از استیت q به آن میرسیم را در نظر بگیرید. این استیت یک استیت accepting

است چون پس از خواندن دنباله $k_i, k_{i+1} \dots k_{m+1} b^{i-1}$ به آن رسیدیم و $1+m$ حرف از آخر برابر a است. از

طرف این استیت یک استیت non-accepting است چون با خواندن دنباله $n_i, n_{i+1} \dots n_{m+1} b^{i-1}$ به آن میرسیم و

$1+m$ حرف از آخر برابر b است. تناقض حاصل نشان میدهد که DFA کمینه این زیان حداقل 2^{m+1} استیت دارد.

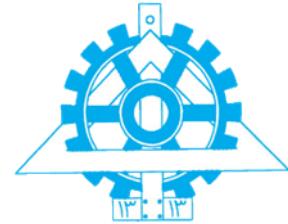


به نام خدا

نظریه زبان ها و ماشین ها - بهار ۱۴۰۱

پاسخ تمرین شماره سه

:ایمیل دستیار آموزشی این مجموعه



f24moh@gmail.com

- در مواردی که خواسته شده نامنظم بودن یا نبودن زبان ها را اثبات کنید، نامنظم بودن هیچ زبانی را به صورت پیش فرض در نظر نگیرید.
- یکی از اهداف این تمرین یادگیری "الم تزریق" میباشد، در صورتی که از روش دیگری برای نشان دادن نامنظم بودن زبانها استفاده کنید نمره ای به حل شما تعلق نمیگیرد.
- برای نشان دادن منظم بودن زبان ها کافی است DFA آنرا رسم کنید و توجه کنید DFA به صورت کمینه شده تنها قابل قبول است.

(1) نامنظم بودن زبان های زیر را اثبات کنید. (25 نمره)

a) $L_1 = \{0^i 1^j \mid \gcd(i, j) = 1\}$ (9 نمره)

b) $L_2 = \{(ab)^n a^k : n > k, k \geq 0\}$ (8 نمره)

c) $L_3 = \{w \mid w \in \{a, b, c\}^* \text{ and } n_a(w) \leq n_b(w) \leq n_c(w)\}$ (8 نمره)

(2) منظم بودن یا نامنظم بودن زبان های زیر را مشخص کنید. (پاسخ خود را اثبات کنید). (24 نمره)

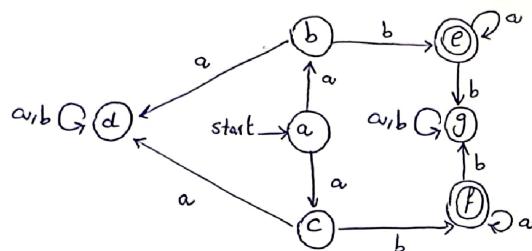
a) $L_1 = \{w_1 w_2 \mid w_1, w_2 \in \{0, 1\}^* \text{ and } |w_1| = 2|w_2|\}$

b) $L_2 = \{a^n b^m c^k \mid n = m \text{ and } n \neq k\}$

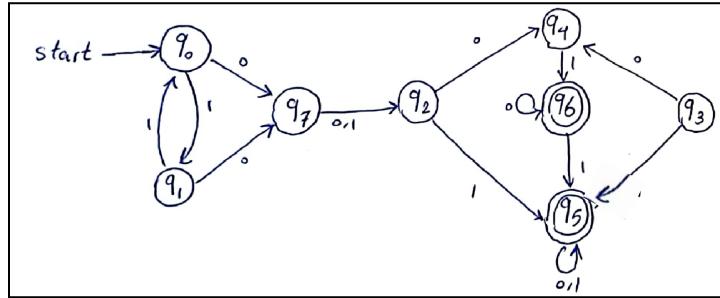
c) $L_3 = \{w \mid w \in \{a, b, c\}^*, |w| \text{ is prime number}\}$

(3) DFA های داده شده را کمینه کنید. (27 نمره)

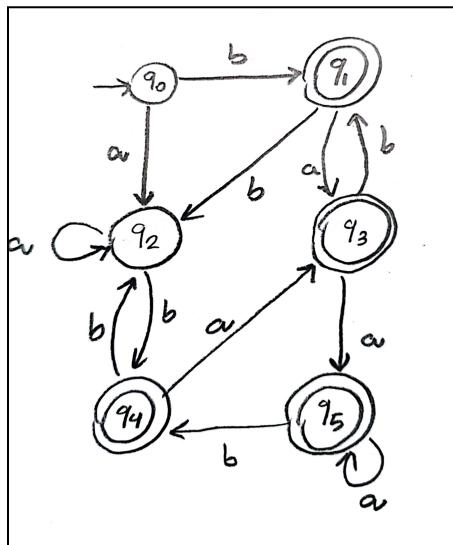
a) $\Sigma = \{a, b\}$



b) $\Sigma = \{0, 1\}$



c) $\Sigma = \{a, b\}$



(4) یک دانشجو علاقمند به مباحث تئوری علوم کامپیوتر با استفاده از لم تزریق میخواهد اثبات کند که این زبان نامنظم است:

$$L = w_1 w_2 \mid w_i \in \{a, b\}^*, n_a(w_1) = n_b(w_2)\}$$

پاسخ این دانشجو به صورت زیر است:

1. حریف مقدار $p \geq 1$ انتخاب میکند
2. من رشته $w = a^p b^p$ را انتخاب میکنم
3. حریف رشته w را به صورت xyz تقسیم بندی میکند و چون طول xy کمتر p باید باشد، y حتماً به صورت a^p میباشد.
4. اگر من $0 = i$ قرار بدهم، تعداد a ها کمتر از تعداد b ها میشود و دیگر این رشته متعلق به زبان نمیباشد، پس زبان نامنظم است.

یا از نظر شما این استدلال صحیح است؟

اگر خیر اشکال کار کجاست؟

(اگر فکر میکنید زبان منظم است کافی است DFA مربوط به زبان را رسم کنید) (10 نمره)

(5) زبان $L_{p(n)}$ به صورت زیر تعریف میشود، به این صورت که $p(n)$ پک چند جمله ای با ضرایب طبیعی باشد.

$$L_{p(n)} = \{w \mid w = 0^{p(n)}, n > 0, n \notin N\}$$

ثابت کنید زبان $L_{p(n)}$ منظم است اگر و تنها اگر درجه جند جمله ای $p(n)$ برابر صفر یا یک باشد. (14 نمره)

(امتیازی) زبان $L(K)$ را به صورت زیر تعریف میکنیم:

$$L(K) = \{w_1 w_2 \mid w_i \in (0 + 1)^*, |w_2| = m, w_1 \text{ ends with } 0\}$$

اثبات کنید DFA کمینه حداقل 2^{m+1} استیت دارد. (10 نمره)

به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۲

تمرين شماره ۴

دستیار آموزشی این مجموعه: مجید فریدفر

majid.faridfar@gmail.com



پاسخنامه

(1) برای هر کدام از زبان‌های زیر، یک گرامر مستقل از متن بنویسید.

2) $L = \{w \in \{a, b, c\}^* \mid 2n_a(w) = n_b(w) + n_c(w)\}$

پاسخ:

$$S \rightarrow aTTS|TaTS|TTbS|\epsilon$$

$$T \rightarrow b|c$$

a)

زبانی شامل رشته‌هایی شامل حروف a و b ، به طوری که کاراکترهایی که در جایگاه فرد قرار دارند، باهم برابرند و کاراکترهایی که در جایگاه زوج قرار دارند، باهم.

پاسخ:

$$S \rightarrow A|B|C|D$$

$$A \rightarrow Aaa|a|a|\epsilon$$

$$B \rightarrow Bab|b|b|\epsilon$$

$$C \rightarrow Cba|a|a|\epsilon$$

$$D \rightarrow Bbb|b|b|\epsilon$$

b) $b(bc + a) * a(a + b) * c *$

پاسخ:

$$S \rightarrow bAaCD$$

$$A \rightarrow bcA|aA|\epsilon$$

$$C \rightarrow aC|bC|\epsilon$$

$$D \rightarrow cD|\epsilon$$

c) $L = \{a^n b^m \mid n \leq m + 3\}$

پاسخ:

$$S \rightarrow aaaA|aaA|aA|\epsilon$$

$$A \rightarrow aAb|B$$

$$B \rightarrow Bb|\epsilon$$

d) $L = \{w \in \{a, b\}^* \mid 2n_a(w) = n_b(w)\}$

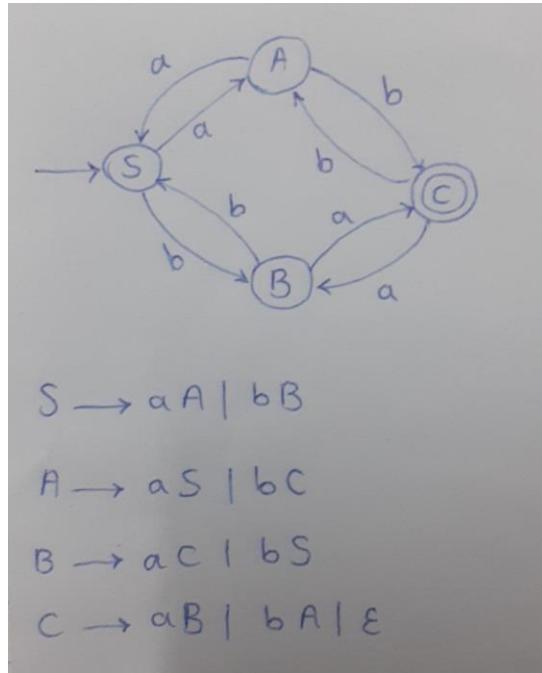
پاسخ:

$$S \rightarrow SbSbSa|SaSbSa|SbSaSa|\epsilon$$

(2) برای زبان‌های زیر، ابتدا اutomaton معادل آن را رسم کنید و سپس گرامر مستقل از متنه را بنویسید.

a) $L = \{w \in \{a, b\}^* \mid n_a(w) * n_b(w) \equiv 1 \pmod{2}\}$

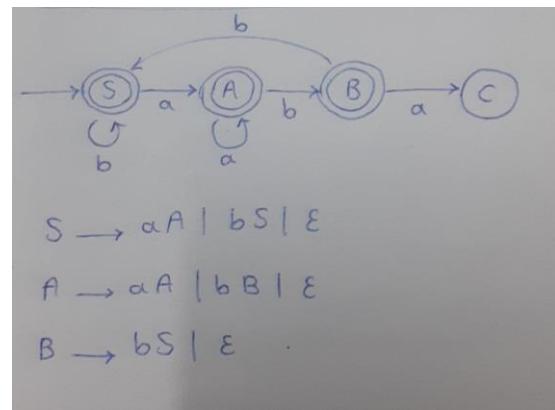
پاسخ: یعنی رشته‌هی که تعداد a ها و تعداد b ها یکسان فرد است.



b)

زبانی که رشته‌هایش، شامل aba نمی‌شوند.

پاسخ:

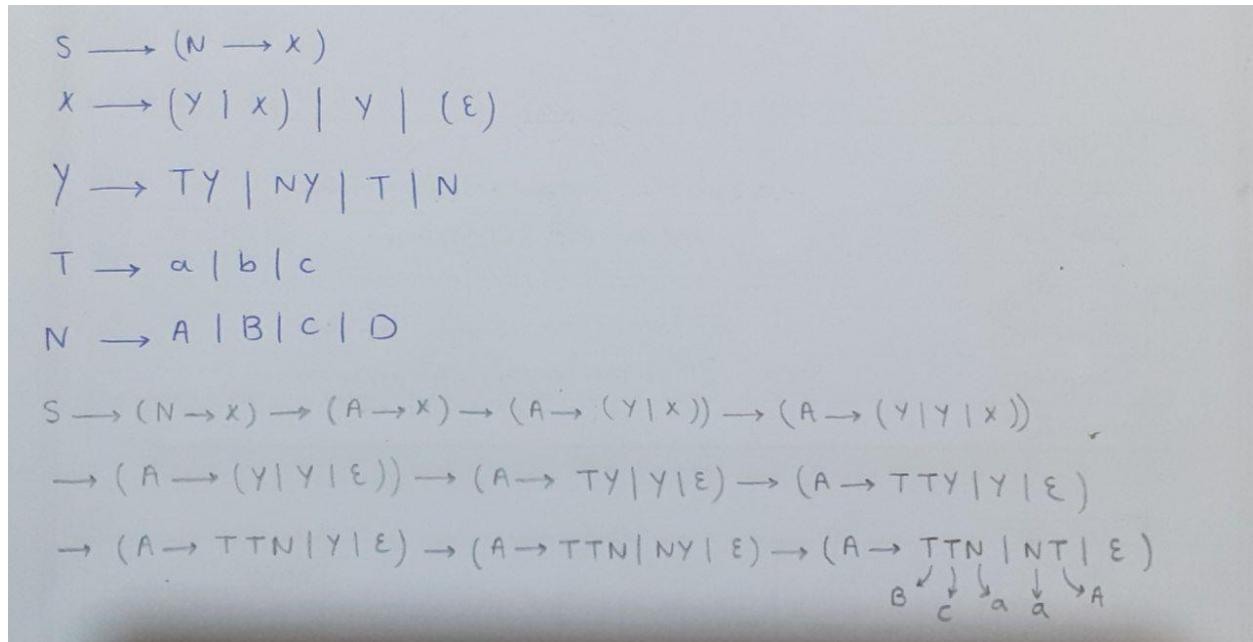


گرامری بنویسید که تمام production rule‌های گرامری را تولید بکند که شامل terminal‌های a و b و c و ϵ و d همین طور non-terminal‌های A و B و C و D می‌شود.

سپس تمام مراحل اشتقاق رشته‌ی زیر را با گرامری که نوشته‌اید، بنویسید.

$$A \rightarrow BCa \mid aA \mid \epsilon$$

پاسخ:



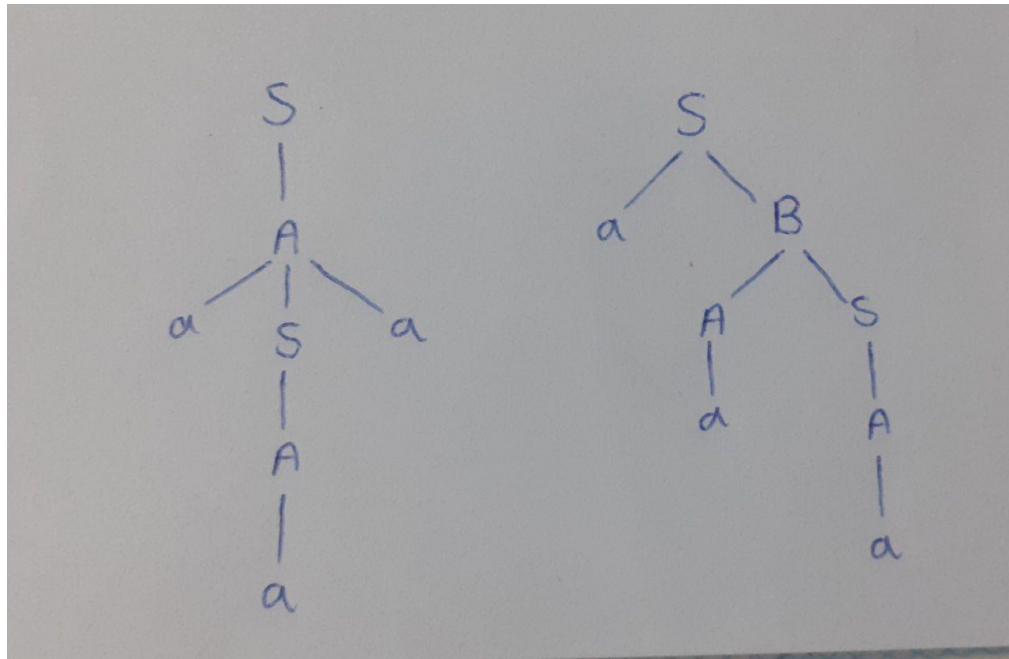
(4) گرامر زیر را در نظر بگیرید:

$$\begin{aligned} S &\rightarrow A \mid aB \\ A &\rightarrow aSa \mid SB \mid a \\ B &\rightarrow AS \end{aligned}$$

a) نشان دهید که این گرامر ابهام دارد. (یعنی رشته‌ای پیدا کنید که برای آن دو درخت اشتقاق مختلف به دست می‌آید).

پاسخ:

این قسمت جواب یکتایی ندارد. اما برای مثال، این گرامر برای رشته‌ی aaa دو تا درخت اشتقاق دارد:



(b) آیا این گزاره صحیح است؟

با توجه به این که زبان این گرامر معادل زبان aa^* است، پس می‌توان گرامر را به این صورت بازنویسی کرد:

$$S \rightarrow aS|a$$

پاسخ:

خیر. این گزاره صحیح نیست. در واقع، زبان این گرامر معادل aa^* نیست. چون برای مثل aa را نمی‌پذیرد.

(c) گرامر $c \rightarrow aS|aSbS|c$ را رفع ابهام کنید (نیازی به اثبات مبهم بودن آن نیست).

پاسخ:

$$S \rightarrow T|R$$

$$T \rightarrow aTbT|c$$

$$R \rightarrow aS|aTbR$$

(5) الگوریتمی برای تبدیل گرامر خطی چپ به گرامر خطی راست ارائه دهید.

پاسخ:

اگر گرامر خطی چپ، قانونی دارد که سمت راست آن با متغیر S شروع می‌شود، قانون زیر را به گرامر اضافه می‌کنیم:

$$S' \rightarrow S$$

به این صورت، S' non-terminal شروع، S خواهد بود.

فرض کنید، A و B و p terminal و non-terminal است. حالا تا جایی که ممکن است، مراحل زیر را انجام می‌دهیم:

۱. اگر گرامر قانونی به شکل $p \rightarrow S$ دارد، آن را نگه می‌داریم.
۲. اگر گرامر قانونی به شکل $A \rightarrow pA$ دارد، قانون $pA \rightarrow S$ را با آن جاگزین می‌کنیم.
۳. اگر گرامر قانونی به شکل $Ap \rightarrow pB$ دارد، قانون $pB \rightarrow A$ را با آن جایگزین می‌کنیم.
۴. اگر گرامر قانونی به شکل $p \rightarrow Ap$ دارد، قانون $p \rightarrow A$ را با آن جایگزین می‌کنیم.

درنهایت، گرامر حاصل، خطی راست خواهد بود.

(6) (امتیازی) فرض کنید گرامر زیر، زبان L_2 را توصیف می‌کند:

$$A \rightarrow bAA|AbA|AAb|a$$

همچنین زبان L_1 به شکل زیر تعریف می‌شود:

$$L_1 = \{w \in \{a, b\}^* \mid n_a(w) = n_b(w) + 1\}$$

ثابت کنید:

$$L_1 = L_2$$

پاسخ:

اثبات دو طرفه است. یعنی باید دو مورد زیر را اثبات کنیم.

۱. $L_1 \subseteq L_2$. هر رشته‌ای که گرامر داده شده تولید می‌کند، متعلق به زبان L_1 است. به عبارت دیگر هر رشته‌ای که این گرامر تولید می‌کند، تعداد a هایش یکی بیشتر از تعداد b هایش است.

۲. $L_2 \subseteq L_1$. گرامر داده شده توانایی تولید تمام رشته‌های زبان L_1 را دارد. به عبارت دیگر تمام رشته‌هایی که تعداد a هایشان یکی بیشتر از تعداد b هایشان است، توسط این گرامر تولید می‌شوند.

در ادامه با استقراری قوی، هر دو موضوع بالا را اثبات می‌کنیم.

پایه: حکم برای رشته‌ای به طول 1 به وضوح برقرار است. (a)

فرض: رشته‌هایی به طول 1 تا $2k-1$ که از گرامر به دست می‌آیند، تعداد a هایشان، یکی بیشتر از تعداد b هایشان است (I). همین طور تمام رشته‌هایی که تعداد a هایشان دقیقاً یکی بیشتر از تعداد b هایشان است، قابل اشتقاق از این گرامر هستند (II).

حکم:

۱. فرض کنید می‌خواهیم یک رشته به طول $2k+1$ را از این گرامر به دست بیاوریم (w). چون $1 < 2k+1$ ، پس اولین قانونی که باید برای به دست آوردن w استفاده کنیم، یکی از سه تای اول خواهد بود. مثلاً اگر از اولی استفاده کرده باشیم ($A \rightarrow bAA$)، یعنی قرار داده‌ایم: $w = buv$. واضح است که طول u و v کمتر از $2k+1$ است. طبق فرض استقرا (قسمت I)، هر دو (u و v) تعداد a هایشان، دقیقاً یکی بیشتر از b هایشان است (چون هر دو از جنس A رشته‌های متعلق به این گرامر هستند). پس:

$$n_a(w) = n_a(u) + n_a(v) \quad \text{و} \quad n_b(w) = 1 + n_b(u) + n_b(v)$$

در نتیجه تعداد a های w دقیقاً یکی بیشتر از تعداد b هایش است.

۲. یک رشته‌ی دلخواه به طول $2k+1$ در نظر بگیرید که تعداد a هایش دقیقاً یکی بیشتر از تعداد b هایش است (w). می‌خواهیم ثابت کنیم که این رشته از این گرامر به دست می‌آید. دو حالت زیر را در نظر بگیرید:

۱. رشته‌ی w با حرف b شروع می‌شود.

- اگر از کاراکتر دوم در رشته شروع به پیمایش کنیم و در هر جایگاهی که هستیم، اختلاف تعداد a ها و b های خوانده شده از جایگاه دوم را در متغیر x بریزیم، در جایگاه دوم (خانه‌ی اول پیمایش) خواهیم داشت: $0 = x$. هم چنین در نهایت خواهیم داشت: $2 = x$ (پس از رد کردن آخرین کاراکتر). پس حتما در این بین، $1 = x$ می‌شود (چون تغییر x در هر محله یا $+1$ است یا -1). پس می‌توانیم بنویسیم: $w = buv$ به طوری که تفاوت تعداد a ها و b های u و v هر کدام، برابر ۱ است (چون تفاوت تعداد a ها و b های w هم برابر یک بود و u را هم به نحوی پیدا کردیم که این خاصیت را داشته باشد (از کاراکتر دوم تا جایی که $x = 1$ شده بود) پس v هم این ویژگی را دارد). پس از قانون $\rightarrow A$ استفاده می‌کنیم و طبق فرض استقرا (\Rightarrow) می‌توانیم u و v را هر چه که باشند، تولید کنیم. پس رشته‌ی w توسط این گرامر قابل تولید شدن است.
- دقت کنید که اگر حرف آخر هم b باشد، مسئله با استدلالی مشابه ثابت می‌شود (پیمایش از آخر). پس حالت دوم را کامل تر می‌کنیم:

۲. رشته‌ی w با حرف a شروع می‌شود و با حرف a تمام می‌شود اگر حرف دوم b بود، کافی است از قانون $A \rightarrow AbA$ استفاده کنیم. چون داریم: $w = abu$ به طوری که تفاوت تعداد a ها و b های u هم برابر یک است و طبق فرض استقرا (\Rightarrow) چون طول آن برابر $2k-1$ است، می‌توان آن را تولید کرد.
- اگر آخر رشته، ba داشته باشیم هم مسئله با استدلالی مشابه اثبات می‌شود ($w = uba$). پس در ادامه می‌گوییم: حالا حالتی را بررسی می‌کنیم که رشته با aa شروع شده و با aa تمام می‌شود. ثابت می‌کنیم حتماً b ای وجود دارد، مقدار x (این بار مقدار x برابر است با اختلاف a ها و b های خوانده از خانه‌ی اول) در آن جایگاه برابر ۱ باشد (تا از قانون دوم $A \rightarrow AbA$ استفاده کنیم).
 - مقدار x را در سمت چپ ترین b ای رشته در نظر بگیرید. این مقدار طبق فرضی که کرده‌ایم، بزرگ‌تر یا مساوی ۲ است (چون حداقل دو تا a قبل از آن داریم). همین طور در سمت راست ترین b ، مقدار آن کمتر یا مساوی صفر است (چون حداقل دو تا a مانده).
 - همین طور مقدار x در خانه‌ی مربوط به یک b تا b ای بعدی، یا بیشتر می‌شود (حداقل دو تا a ببینیم)، یا ثابت می‌ماند (یک a ببینیم) یا یک واحد کمتر (هیچ a ای نببینیم) می‌شود. پس در این پیمایش از چپ به راست، حتماً به جایی می‌رسیم که مقدار x در یکی از b ها برابر ۱ می‌شود (چون باید از مقداری بیشتر از ۲ به مقداری کمتر از ۰ برسیم). حالا از قانون دوم استفاده کرده و طبق فرض استقرا، رشته‌ی w را می‌سازیم ($w = ubv$).

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۲

تمرین شماره ۴

دستیار آموزشی این مجموعه: مجید فریدفر

majid.faridfar@gmail.com



تاریخ تحولی: ۲۳ فروردین



(1) برای هر کدام از زبان‌های زیر، یک گرامر مستقل از متن بنویسید.

a) $L = \{w \in \{a, b, c\}^* \mid 2n_a(w) = n_b(w) + n_c(w)\}$

b)

زبانی شامل رشته‌هایی شامل حروف a و b، به طوری که کاراکترهایی که در جایگاه فرد قرار دارند، باهم برابرند و کاراکترهایی که در جایگاه زوج قرار دارند، باهم.

c) $b(bc + a) * a(a + b) * c *$

d) $L = \{a^n b^m \mid n \leq m + 3\}$

e) $L = \{w \in \{a, b\}^* \mid 2n_a(w) = n_b(w)\}$

(2) برای زبان‌های زیر، ابتدا اutomaton معادل آن را رسم کنید و سپس گرامر مستقل از متنش را بنویسید.

a) $L = \{w \in \{a, b\}^* \mid n_a(w) * n_b(w) \equiv 1 \pmod{2}\}$

b)

زبانی که رشته‌هایش، شامل aba نمی‌شوند.

(3) گرامری بنویسید که تمام production rule‌های گرامری را تولید بکند که شامل terminal‌های a و b و c و

همین طور non-terminal‌های A و B و C و D می‌شود.

سپس تمام مراحل اشتقاق رشته‌ی زیر را با گرامری که نوشته‌اید، بنویسید.

$$A \rightarrow BCa|aA|\epsilon$$

توضیح بیشتر: برای مثال می‌توان "C → A → " را مثل یک رشته نگاه کرد. که کاراکتر اول آن 'A'، کاراکتر دوم '→'، کاراکتر سوم 'B'، کاراکتر چهارم ']' و کاراکتر پنجم 'C' است. خواسته‌ی سوال این است که گرامری بنویسید که بتواند چنین رشته‌هایی را تولید بکند (که بتوان آن را بعنوان یک قانون در نظر گرفت).

(4) گرامر زیر را در نظر بگیرید:

$$S \rightarrow A|aB$$

$$A \rightarrow aSa|SB|a$$

$$B \rightarrow AS$$

(a) نشان دهید که این گرامر ابهام دارد. (یعنی رشته‌ای پیدا کنید که برای آن دو درخت اشتقاق مختلف به دست می‌آید. این دو درخت را هم رسم کنید).

(b) آیا این گزاره صحیح است؟

با توجه به این که زبان این گرامر معادل زبان aa^* است، پس می‌توان گرامر را به این صورت بازنویسی کرد:

$$S \rightarrow aS|a$$

(c) گرامر $S \rightarrow aS|aSbS|c$ را رفع ابهام کنید (نیازی به اثبات مبهم بودن آن نیست).

(5) الگوریتمی برای تبدیل گرامر خطی چپ به گرامر خطی راست ارائه دهید.

(6) (امتیازی) فرض کنید گرامر زیر، زبان L_2 را توصیف می‌کند:

$$A \rightarrow bAA \mid AbA \mid AAb \mid a$$

همچنین زبان L_1 به شکل زیر تعریف می‌شود:

$$L_1 = \{w \in \{a, b\}^* \mid n_a(w) = n_b(w) + 1\}$$

ثابت کنید:

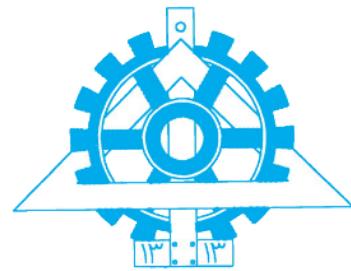
$$L_1 = L_2$$

راهنمایی: روی طول رشته‌ها، استقرای قوی بزنید.



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



پاسخ تمرین شماره ۵

دستیار آموزشی این مجموعه: مریم جعفرآبادی آشتیانی
maryamjafarabadi88@gmail.com
 تاریخ تحویل: ۳۰ فروردین (صفحه درس)

(۱) سمبول‌های D, E اضافه هستند و با حذف آن‌ها داریم:

$$S \rightarrow CA | aAa | bBb | \epsilon$$

$$A \rightarrow C | a$$

$$B \rightarrow C | b$$

$$C \rightarrow b | \epsilon$$

ب) ابتدا قاعده $\epsilon \rightarrow S'$ را اضافه می‌کنیم و قاعده $\epsilon \rightarrow S$ را حذف می‌کنیم.

حذف $\epsilon : C \rightarrow \epsilon$

$$S' \rightarrow S | \epsilon$$

$$S \rightarrow CA | A | aAa | bBb$$

$$A \rightarrow C | a | \epsilon$$

$$B \rightarrow C | b | \epsilon$$

$$C \rightarrow b$$

حذف $\epsilon : A \rightarrow \epsilon, B \rightarrow \epsilon$

$$S \rightarrow CA | C | A | aAa | aa | bBb | bb | \epsilon$$

$$A \rightarrow C | a$$

$$B \rightarrow C | b$$

$$C \rightarrow b$$

قاعده نهایی در پایان قسمت ب)

$$S' \rightarrow S | \varepsilon$$

$$S \rightarrow CA | C | A | aAa | aa | bBb | bb$$

$$A \rightarrow C | a$$

$$B \rightarrow C | b$$

$$C \rightarrow b$$

ج) حذف ($A \rightarrow C, B \rightarrow C$

$$S' \rightarrow S | \varepsilon$$

$$S \rightarrow CA | C | A | aAa | aa | bBb | bb$$

$$A \rightarrow b | a$$

$$B \rightarrow b$$

$$C \rightarrow b$$

حذف ($S \rightarrow A, S \rightarrow C$

$$S' \rightarrow S | \varepsilon$$

$$S \rightarrow CA | b | a | aAa | aa | bBb | bb$$

$$A \rightarrow b | a$$

$$B \rightarrow b$$

$$C \rightarrow b$$

حذف ($S' \rightarrow S$, نتیجه حاصل از قسمت ج)

$$S' \rightarrow CA | b | a | aAa | aa | bBb | bb | \varepsilon$$

$$S \rightarrow CA \mid b \mid a \mid aAa \mid aa \mid bBb \mid bb$$
$$A \rightarrow b \mid a$$
$$B \rightarrow b$$
$$C \rightarrow b$$

د) قاعد C بدون کاربرد و تکراری است.

$$S' \rightarrow BA \mid b \mid a \mid aAa \mid aa \mid bBb \mid bb \mid \epsilon$$
$$S \rightarrow BA \mid b \mid a \mid aAa \mid aa \mid bBb \mid bb$$
$$A \rightarrow b \mid a$$
$$B \rightarrow b$$

ه) فرم نرمال چامسکی :

$$S' \rightarrow BA \mid b \mid a \mid A'AA' \mid A'A' \mid BBB \mid BB \mid \epsilon$$
$$S \rightarrow BA \mid b \mid a \mid A'AA' \mid A'A' \mid BBB \mid BB$$
$$A \rightarrow b \mid a$$
$$B \rightarrow b$$
$$A' \rightarrow a$$

نتیجه قسمت ه)

$$S' \rightarrow BA \mid b \mid a \mid CA' \mid A'A' \mid DB \mid BB \mid \epsilon$$
$$S \rightarrow BA \mid b \mid a \mid CA' \mid A'A' \mid DB \mid BB$$
$$C \rightarrow A'A$$

$D \rightarrow BB$

$A \rightarrow b | a$

$B \rightarrow b$

$A' \rightarrow a$

.2

این جمله نادرست میباشد چرا که با استفاده از این گرامر به دلیل اینکه قاعده اپسیلون نداریم و در هر production rule تعداد ترمینال و نان ترمینال تولید شده عددی فرد است، نمیتوانیم رشته هایی به طول زوج را ایجاد کنیم. به عنوان مثال نمیتوانیم گرامری به فرم بالا بنویسیم برای پذیرش .aabb

.3

ترتیبی برای رفع چپگردی در نظرمیگیریم (در اینجا SAB)

: حذف چپگردی مستقیم S

$S \rightarrow ABS' | cS' | AB | c$

$S' \rightarrow AS' | BS' | A | B$

$A \rightarrow SSA | B | a$

$B \rightarrow Bb | d$

: تبدیل چپگردی غیر مستقیم A به مستقیم

$S \rightarrow ABS' | cS' | AB | c$

$S' \rightarrow AS' | BS' | A | B$

$A \rightarrow ABS'SA | cS'SA | ABSA | cSA | B | a$

$B \rightarrow Bb | d$

: حذف چپگردی مستقیم A

$$S \rightarrow ABS' \mid cS' \mid AB \mid c$$

$$S' \rightarrow AS' \mid BS' \mid A \mid B$$

$$A \rightarrow cS'SAA' \mid cSAA' \mid BA' \mid aA' \mid cS'SA \mid cSA \mid B \mid a$$

$$A' \rightarrow BS'SAA' \mid BSAA' \mid BS'SA \mid BSA$$

$$B \rightarrow Bb \mid d$$

حذف چیگردی مستقیم : B

$$S \rightarrow ABS' \mid cS' \mid AB \mid c$$

$$S' \rightarrow AS' \mid BS' \mid A \mid B$$

$$A \rightarrow cS'SAA' \mid cSAA' \mid BA' \mid aA' \mid cS'SA \mid cSA \mid B \mid a$$

$$A' \rightarrow BS'SAA' \mid BSAA' \mid BS'SA \mid BSA$$

$$B \rightarrow dB' \mid d$$

$$B' \rightarrow bB' \mid b$$

.4

S					
D	C				
S	S	—			
—	C	—	C		
—	S	—	S	S	
A	A	B	B	A	B

همانطور که میبینیم در راس جدول S را داریم که نشان دهنده این است که رشته پذیرفته میشود.

5. گرامر زیر را به فرم نرمال گریباخ تبدیل کنید. (20 نمره)

$$E \rightarrow E + T | T$$

$$T \rightarrow T * F | F$$

$$F \rightarrow (E) | a$$

ابتدا باید قواعد اپسیلون و یکه را در صورت وجود حذف کنیم و رفع چیگردی کنیم.
قواعد اپسیلون وجود ندارد ولی پس از حذف قواعد یکه داریم:

$$E \rightarrow E + T | T * F | (E) | a$$

$$T \rightarrow T * F | (E) | a$$

$$F \rightarrow (E) | a$$

رفع چیگردی :

$$E \rightarrow T * FE' | (E)E' | aE' | T * F | (E) | a$$

$$E' \rightarrow + TE' | + T$$

$$T \rightarrow (E)T' | aT' | (E) | a$$

$$T' \rightarrow * FT' | * F$$

$$F \rightarrow (E) | a$$

تبدیل به فرم نرمال گریباخ:

آنقدر از سمت چپ گرامر را گسترش میدهیم که به یک ترمینال بررسیم:

$$E \rightarrow (E)T' * FE' | aT' * FE' | a * FE' | (E)E' | aE' | (E)T' * F | aT' * F | a * F | (E) | a$$

$$E' \rightarrow + TE' | + T$$

$$T \rightarrow (E)T' | aT' | (E) | a$$

$$T' \rightarrow * FT' | * F$$

$$F \rightarrow (E) | a$$

حال ترمینال های میانی را با نان ترمینال جایگزین میکنیم:

$E \rightarrow (EPT'MFE' \mid aT'MFE' \mid aMFE' \mid (EPE' \mid aE' \mid (EPT'MF \mid aT'MF \mid aMF \mid (EP \mid a$

$E' \rightarrow + TE' \mid + T$

$T \rightarrow (EPT' \mid aT' \mid (EP \mid a$

$T' \rightarrow * FT' \mid * F$

$F \rightarrow (EP \mid a$

$P \rightarrow)$

$M \rightarrow *$

6. (امتیازی) اگر G یک گرامر مستقل از متن باشد و W رشته‌ای به طول l باشد که متعلق به زبان گرامر G است. حال اگر G به فرم نرمال چامسکی باشد، طول اشتقاق رشته W چقدر است؟ توضیح دهید چرا.(10 نمره)

طول این اشتقاق معادل $1 - 2l$ میباشد. چراکه در گرامری به فرم نرمال چامسکی قواعد به یکی از دو صورت

زیر میباشند:

$A \rightarrow BC$

$A \rightarrow a$

پس برای ساخت رشته W ابتدا باید تعدادی نان ترمینال تولید کنیم به تعداد l . که برای این کار باید از $1 - l$ قاعده

به فرم اول استفاده کنیم و سپس این نان ترمینال‌ها با استفاده از قواعد از نوع دوم ترمینال ایجاد میکنند. که باید از l

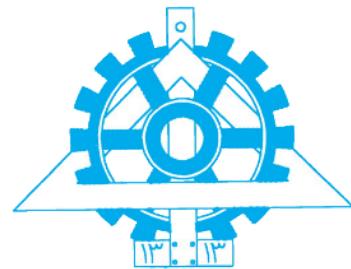
قاعده هم برای تبدیل نان ترمینال‌ها به ترمینال استفاده کنیم. پس در مجموع برای رسیدن به رشته W نیاز است از

$1 - 2l$ قاعده استفاده شود که بدین معنیست که طول اشتقاق این رشته $1 - 2l$ میباشد.



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۵

دستیار آموزشی این مجموعه: مریم جعفرآبادی آشتیانی
maryamjafarabadi88@gmail.com
تاریخ تحویل: ۳۰ فروردین (صفحه درس)

۱. گرامر زیر را در نظر بگیرید و موارد خواسته شده را روی آن اعمال کنید: (۲۵ نمره)

$$S \rightarrow AE \mid CA \mid aAa \mid bBb \mid \epsilon$$

$$A \rightarrow C \mid a$$

$$B \rightarrow C \mid b$$

$$C \rightarrow aE \mid b \mid \epsilon$$

$$D \rightarrow A \mid B \mid ab$$

$$E \rightarrow EC \mid AE$$

الف) سمبول‌های بدون کاربرد را (در صورت وجود) حذف کنید.

ب) قواعد اپسیلون را حذف کنید.

ج) قواعد یکه را حذف کنید.

د) همچنان در صورت وجود، قواعد بی کاربرد را حذف کنید.

ه) گرامر معادل به دست آمده را به فرم نرمال چامسکی تبدیل کنید.

۲. جمله زیر را در صورت درست بودن اثبات کنید و در غیر این صورت مثال نقض بیاورید (ادعای خود را به طور کامل توضیح دهید). (۱۵ نمره)

برای هر گرامر مستقل از متن دلخواهی که قاعده اپسیلون در آن نباشد میتوان یک گرامر معادل که در آن قواعده‌ش به یکی از دو فرم زیر باشد نوشت.

$$A \rightarrow BCD$$

$$A \rightarrow a$$

که متغیر های B, C, D همه غیر ترمینال هستند و a ترمینال میباشد.

3. چیگردنی را از گرامر زیر حذف کنید. (20 نمره)

$$S \rightarrow SA \mid SB \mid AB \mid c$$

$$A \rightarrow SSA \mid B \mid a$$

$$B \rightarrow Bb \mid d$$

4. الگوریتم CYK را روی گرامر زیر اجرا کنید و با رسم جدول نشان دهید که آیا رشته $"aabbaab"$ پذیرفته میشود یا خیر. (20 نمره)

$$S \rightarrow AB \mid BA \mid SS \mid AC \mid BD$$

$$A \rightarrow a$$

$$B \rightarrow b$$

$$C \rightarrow SB$$

$$D \rightarrow SA$$

5. گرامر زیر را به فرم نرمال گریباخ تبدیل کنید. (20 نمره)

$$E \rightarrow E + T \mid T$$

$$T \rightarrow T * F \mid F$$

$$F \rightarrow (E) \mid a$$

6. (امتیازی) اگر G یک گرامر مستقل از متن باشد و W رشته‌ای به طول l باشد که متعلق به زبان گرامر G است. حال اگر G به فرم نرمال چامسکی باشد، طول اشتقاق رشته W چقدر است؟ توضیح دهید چرا.(10 نمره)

به نام خدا



نظريه زبانها و ماشينها - بهار ۱۴۰۲

تمرین شماره ۶

دستيary آموزشی اين مجموعه: پاشا براهييمى
pashabarahimi@gmail.com



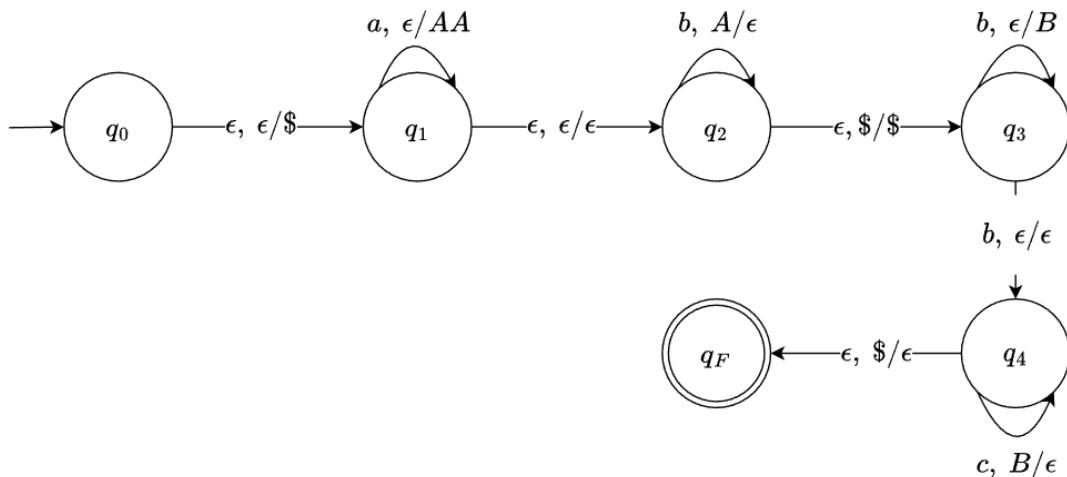
تاریخ تحويل: ۶ اردیبهشت (صفحه درس)

نکته: در تمامی سوالاتی که باید PDA رسم کنید، فرض کنید \$ در استک قرار ندارد و در صورت نیاز باید خودتان این مورد را در استک push کنید.

(1) برای هر کدام از زبان‌های زیر یک PDA رسم کنید. (20 نمره)

الف) $L_1 = \{a^i b^j c^k \mid j = 2i + k + 1; i, j, k \geq 0\}$

پاسخ:



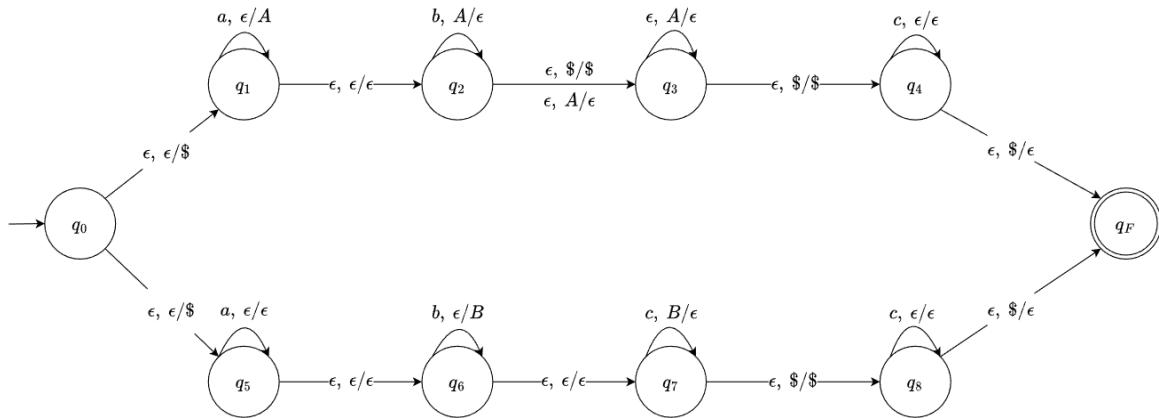
ب) $L_2 = \{a^i b^j c^k \mid j \leq \max(i, k); i, j, k \geq 0\}$

پاسخ:

در این بخش اجتماع دو زبان را خواهیم داشت:

1- حالتی که تعداد b از تعداد a بیشتر نباشد.

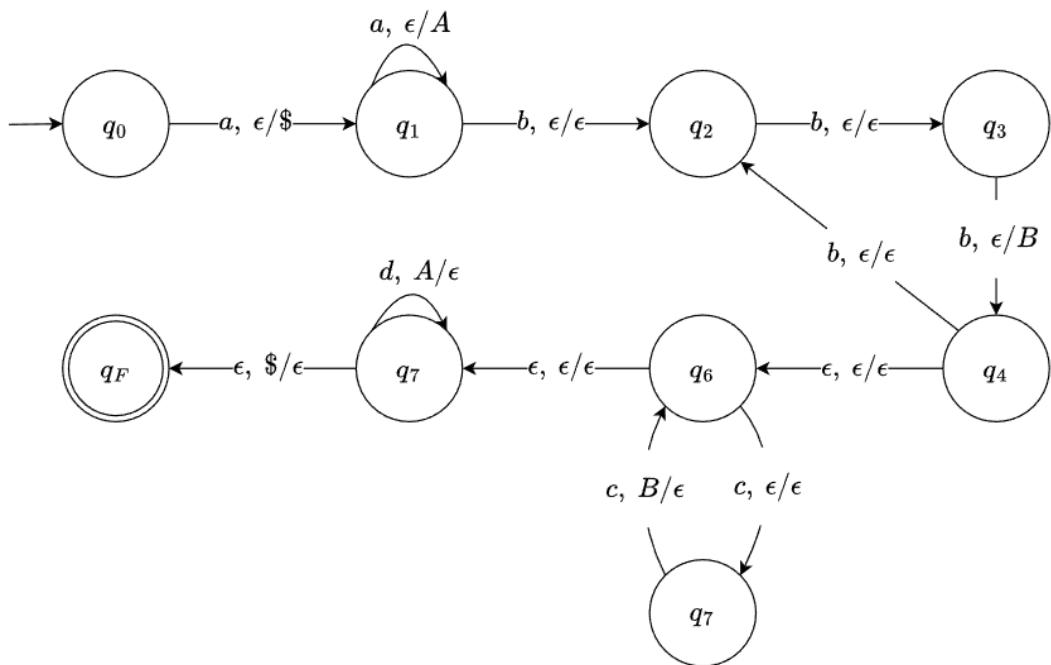
2- حالتی که تعداد b از تعداد c بیشتر نباشد.



$$ج) L_3 = \{a^n b^{3m} c^{2m} d^{n-1} \mid n, m \geq 1\}$$

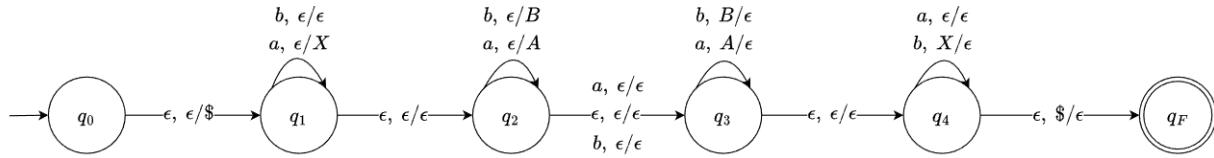
پاسخ:

لازم به ذکر است که PDA شما نباید رشته‌ای را بپذیرد که تعداد a یا b یا c آن برابر با 0 است.



$$د) L_4 = \{xwy \mid x, y, w \in \{a, b\}^* \text{ and } w = w^R \text{ and } n_a(x) = n_b(y)\}$$

پاسخ:



(2) برای گرامر زیر یک PDA با حداکثر 3 حالت^۱ رسم کنید و سپس زبان گرامر را بنویسید. (15 نمره)

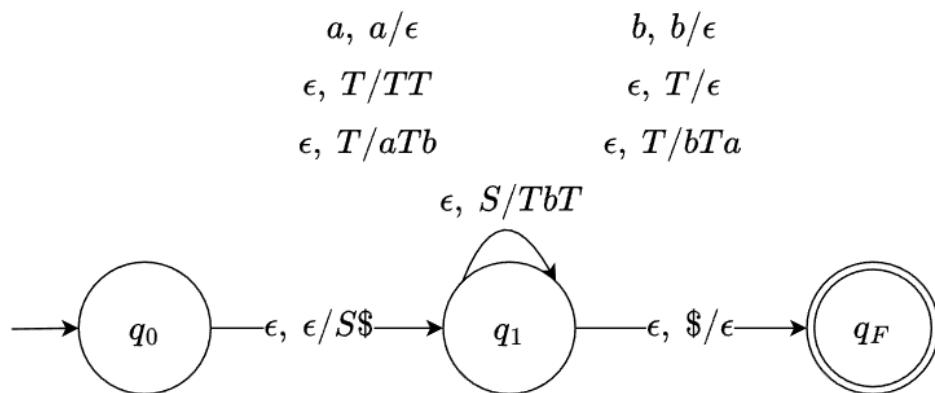
$$S \rightarrow TbT$$

$$T \rightarrow aTb \mid bTa \mid TT \mid \epsilon$$

پاسخ:

زبانی که این گرامر می‌پذیرد برابر است با رشته‌هایی شامل a و b که تعداد b آنها، دقیقاً یکی بیشتر از تعداد a آنها است، یا به عبارت دیگر:

$$L = \{w \mid w \in \{a, b\}^* \text{ and } n_b(w) = n_a(w) + 1\}$$



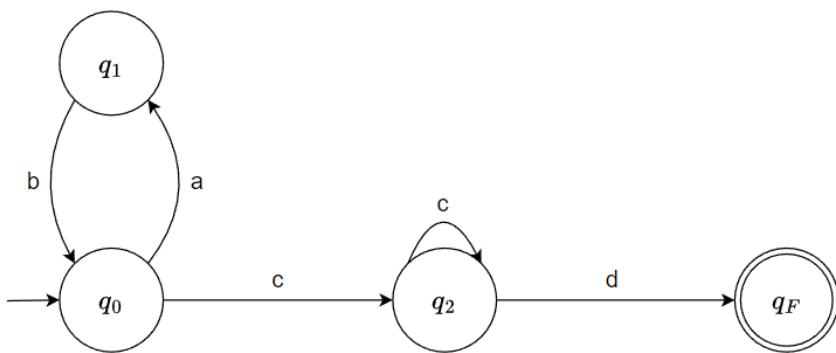
(3) زبان زیر را در نظر بگیرید: (20 نمره)

$$L = \{w \mid w = (ab)^* cc^* d\}$$

^۱ State

- الف) برای این زبان یک NFA رسم کنید.
- ب) آیا می‌توانید NFA بخش قبل را با فقط 3 حالت رسم کنید؟ در صورت امکان این کار را انجام دهید و در غیر این صورت، دلیل خود را به صورت خلاصه ذکر کنید.
- ج) آیا می‌توانید یک PDA برای این زبان با حداقل 3 حالت رسم کنید؟ در صورت امکان این کار را انجام دهید و در غیر این صورت، دلیل خود را به صورت خلاصه توضیح دهید.
- د) آیا می‌توان برای هر زبان Context-Free یک PDA با 3 حالت رسم کرد؟ در صورت امکان، روش کار خود را توضیح دهید.

پاسخ:



این امکان وجود ندارد که NFA را با کمتر از 4 حالت رسم کنیم، زیرا در NFA تنها روشی که برای ذخیره کردن state فعلی داریم، اضافه کردن حالت به NFA است و در این بخش، 4 حالت متفاوت خواهیم داشت:

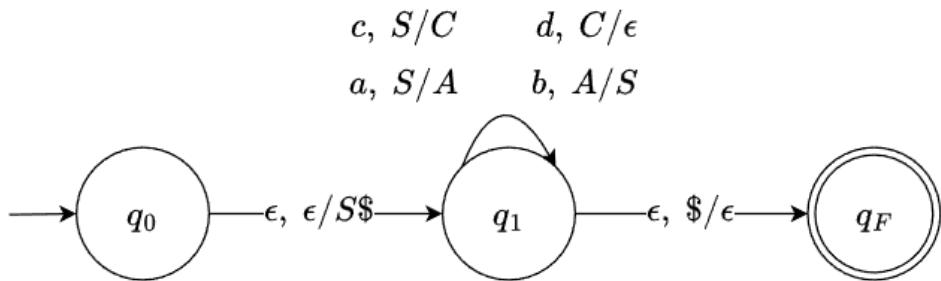
1- حالتی که یک a دیده‌ایم اما بعد از این a، هیچ b-ای ندیده‌ایم.

2- حالتی که به تعداد دلخواه ab دیده‌ایم.

3- حالتی که بعد از مشاهده تعداد دلخواه ab، حداقل یک c دیده‌ایم.

4- حالتی که یک d نیز بعد از حالت سوم دیده‌ایم و به accepting state رسیده‌ایم.

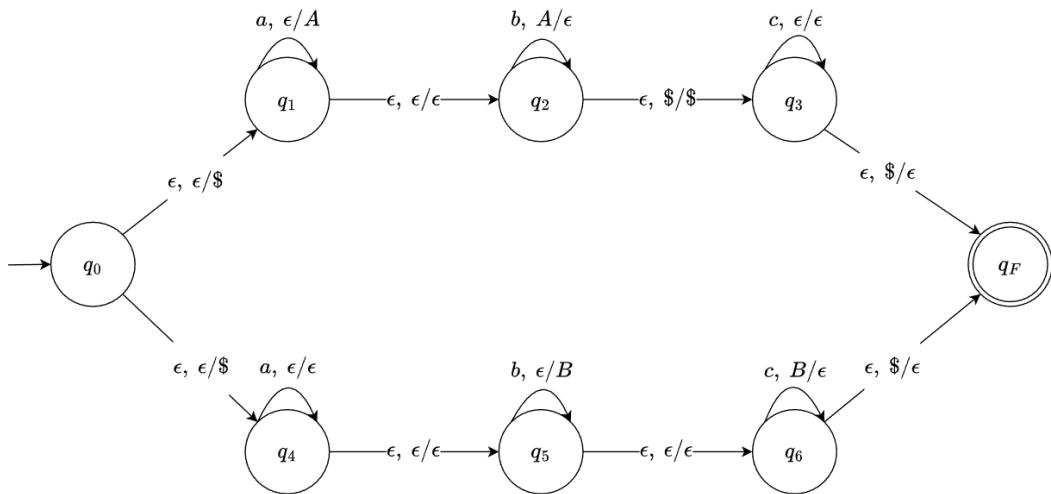
اما در PDA می‌توانیم حالت را جای اضافه کردن state به PDA، در استک ذخیره کنیم. به همین دلیل است که می‌توانیم این زبان را در PDA با تعداد حالت کمتر نشان دهیم.



هر زبان Context-Free را می‌توان با یک PDA با 3 حالت نشان داد. برای این کار ابتدا یک گرامر برای زبان می‌نویسیم. در ابتدای PDA یک S (سمبل شروع) را در استک push می‌کنیم و سپس به ازای هر قانون $A_i \rightarrow \lambda_i$ (سمت چپ غیر پایانه است)، یک حلقه به صورت $\epsilon, A_i/\lambda_i$ روی حالت دوم قرار می‌دهیم. در نهایت به ازای هر پایانه a موجود در الفبا، یک حلقه $\epsilon, a/\epsilon$ روی حالت دوم قرار می‌دهیم. تمامی مراحل مشابه پاسخ سوال 2 است.

(4) زبانی که هر کدام از A های زیر می‌پذیرد را بنویسید. (20 نمره)

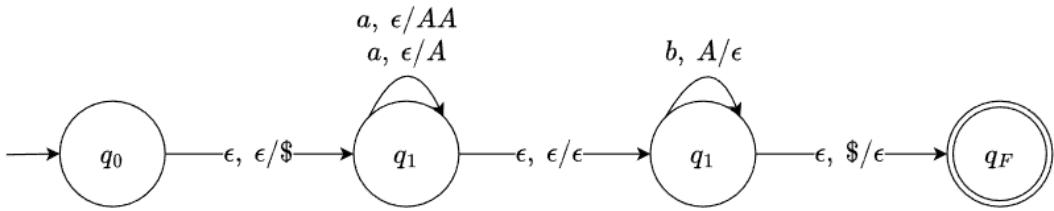
(الف)



پاسخ:

$$L = \{a^i b^j c^k \mid i, j, k \geq 0; i = j \text{ or } j = k\}$$

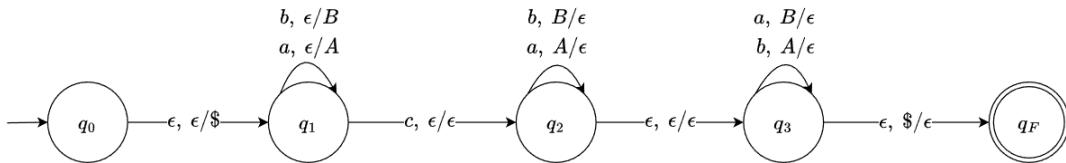
(ب)



پاسخ:

$$L = \{a^n b^m \mid n \leq m \leq 2n\}$$

(5) گرامر متناظر با PDA زیر را بنویسید. (15 نمره)



پاسخ:

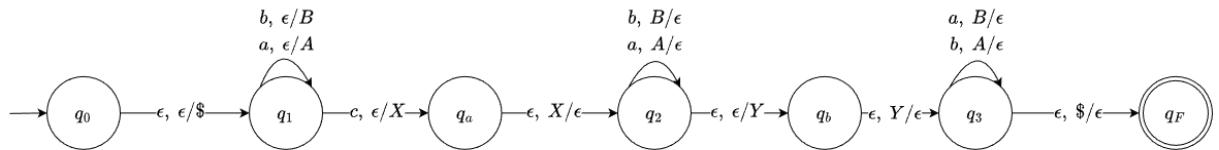
$$S \rightarrow aSb \mid bSa \mid T$$

$$T \rightarrow aTa \mid bTb \mid c$$

روش دوم:

ابتدا باید این PDA را به Simplified PDA تبدیل کنیم. در این نوع PDA شرایط زیر برقرار است:

1. فقط یک accepting state داریم.
2. قبل از قبول کردن رشته stack را خالی می‌کند.
3. هر گذار شامل دقیقاً یک push و یا یک pop است ولی شامل هر 2 نیست.



$A_{00} \rightarrow \epsilon$	$A_{01} \rightarrow A_{01}A_{11}$	$A_{11} \rightarrow A_{11}A_{11}$	$A_{a2} \rightarrow A_{a2}A_{22}$	$A_{b3} \rightarrow A_{b3}A_{33}$	$A_{0F} \rightarrow A_{13}$
$A_{11} \rightarrow \epsilon$	$A_{0a} \rightarrow A_{01}A_{1a}$	$A_{1a} \rightarrow A_{11}A_{1a}$	$A_{ab} \rightarrow A_{a2}A_{2b}$	$A_{bF} \rightarrow A_{b3}A_{3F}$	$A_{12} \rightarrow bA_{12}b$
$A_{aa} \rightarrow \epsilon$	$A_{02} \rightarrow A_{01}A_{12}$	$A_{12} \rightarrow A_{11}A_{12}$	$A_{a3} \rightarrow A_{a2}A_{23}$	$A_{33} \rightarrow A_{33}A_{33}$	$A_{12} \rightarrow aA_{12}a$
$A_{22} \rightarrow \epsilon$	$A_{02} \rightarrow A_{0a}A_{a2}$	$A_{12} \rightarrow A_{1a}A_{a2}$	$A_{a3} \rightarrow A_{ab}A_{b3}$	$A_{3F} \rightarrow A_{33}A_{3F}$	$A_{13} \rightarrow bA_{13}a$
$A_{bb} \rightarrow \epsilon$	$A_{02} \rightarrow A_{02}A_{22}$	$A_{12} \rightarrow A_{12}A_{22}$	$A_{a3} \rightarrow A_{a3}A_{33}$		$A_{13} \rightarrow aA_{13}b$
$A_{33} \rightarrow \epsilon$	$A_{0b} \rightarrow A_{01}A_{1b}$	$A_{1b} \rightarrow A_{12}A_{2b}$	$A_{22} \rightarrow A_{22}A_{22}$		$A_{12} \rightarrow cA_{aa}$
$A_{FF} \rightarrow \epsilon$	$A_{0b} \rightarrow A_{0a}A_{ab}$ $A_{0b} \rightarrow A_{02}A_{2b}$ $A_{03} \rightarrow A_{01}A_{13}$ $A_{03} \rightarrow A_{0a}A_{a3}$ $A_{03} \rightarrow A_{02}A_{23}$ $A_{03} \rightarrow A_{0b}A_{b3}$ $A_{03} \rightarrow A_{03}A_{33}$ $A_{0F} \rightarrow A_{01}A_{1F}$ $A_{0F} \rightarrow A_{0a}A_{aF}$ $A_{0F} \rightarrow A_{02}A_{2F}$ $A_{0F} \rightarrow A_{0b}A_{bF}$ $A_{0F} \rightarrow A_{03}A_{3F}$	$A_{13} \rightarrow A_{11}A_{13}$ $A_{13} \rightarrow A_{1a}A_{a3}$ $A_{13} \rightarrow A_{12}A_{23}$ $A_{13} \rightarrow A_{1b}A_{b3}$ $A_{13} \rightarrow A_{13}A_{33}$ $A_{1F} \rightarrow A_{11}A_{1F}$ $A_{1F} \rightarrow A_{1a}A_{aF}$ $A_{1F} \rightarrow A_{12}A_{2F}$ $A_{1F} \rightarrow A_{1b}A_{bF}$ $A_{1F} \rightarrow A_{13}A_{3F}$	$A_{2b} \rightarrow A_{22}A_{2b}$ $A_{23} \rightarrow A_{22}A_{23}$ $A_{23} \rightarrow A_{2b}A_{b3}$ $A_{23} \rightarrow A_{23}A_{33}$ $A_{2F} \rightarrow A_{22}A_{2F}$ $A_{2F} \rightarrow A_{2b}A_{bF}$ $A_{2F} \rightarrow A_{23}A_{3F}$	$A_{23} \rightarrow A_{22}$	

(6) آیا زبان زیر مستقل از متن است؟ پاسخ خود را اثبات کنید. (10 نمره)

$$L = \{a^n b^m \mid n \neq 3m + 1, n, m \geq 1\}$$

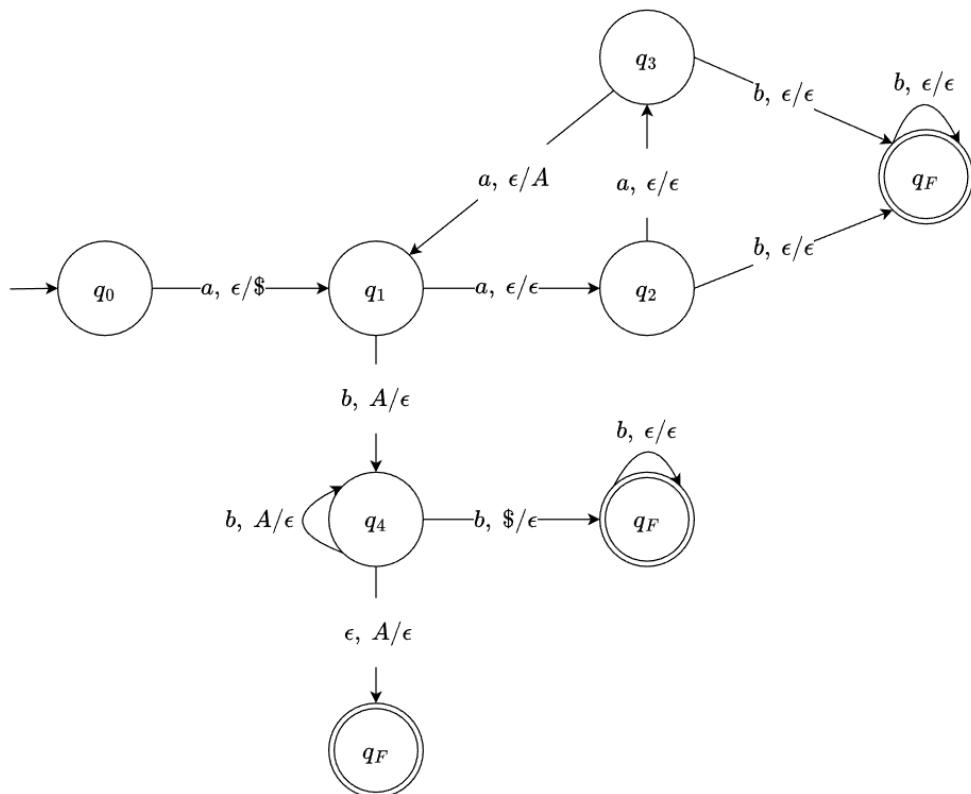
پاسخ:

بله مستقل از متن است. برای اثبات کافیست یک PDA برای این زبان رسم کنیم.

برای این سوال 3 حالت را در نظر می‌گیریم:

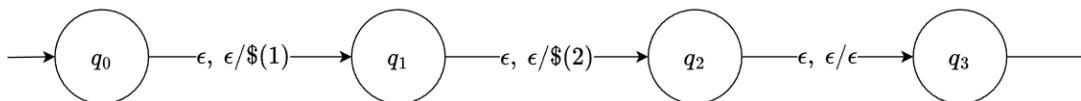
1. $n = 3k$
2. $n = 3k + 1$
3. $n = 3k + 2$

تنها حالتی که ممکن است مشکل ایجاد کند، حالت دوم است. در نتیجه می‌توانیم این 3 حالت را از هم جدا کنیم:



(7) امتیازی: به PDA با 2 استک Double-Stack PDA گفته می‌شود. در این PDA شما به 2 عدد استک دسترسی دارید و در هر گذار² می‌توانید فقط در یک استک عملیات انجام دهید و در هر عملیات مشخص می‌کنید که از کدام استک استفاده می‌کنید. (10 نمره)

برای رسم این PDA به مثال زیر توجه کنید:



در این مثال توسط گذار اول \$ در استک اول push می‌شود و در گذار دوم \$ در استک دوم push می‌شود. در گذار سوم به دلیل عدم انجام عملیات در استک، هیچ استکی مشخص نشده است.

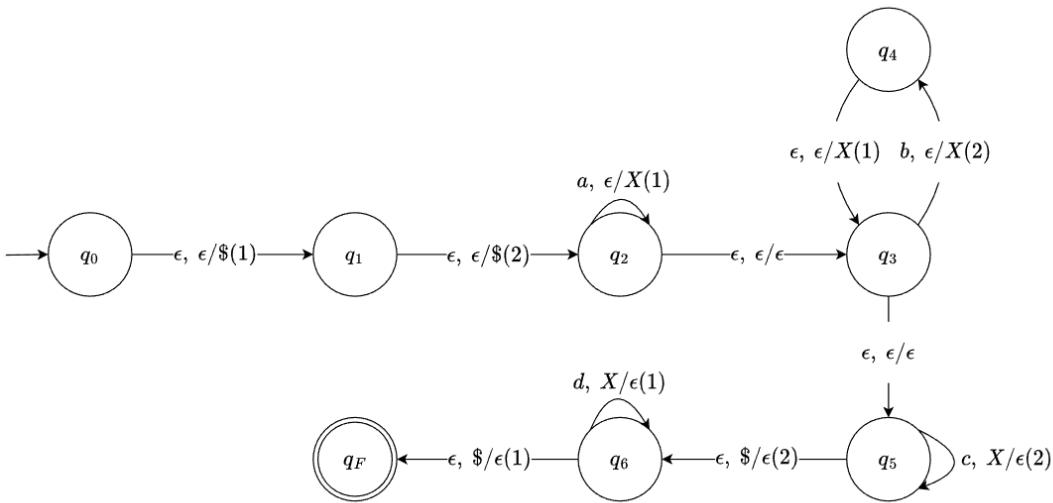
الف) یک PDA برای زبان زیر رسم کنید.

$$L = \{a^n b^m c^m d^{n+m} \mid n, m \geq 0\}$$

ب) آیا قدرت Single-Stack PDA با Double-Stack PDA یکسان است؟ به صورت خلاصه دلیل خود را توضیح دهید. دقت کنید که برای برابر بودن قدرت این دو PDA، لازم است این برابری به صورت دو طرفه وجود داشته باشد؛ در نتیجه این مورد را در دلایل خود (در صورتی که پاسخ شما برابری قدرت است)، لحاظ کنید.

پاسخ:

² Transition



خیر قدرت این دو PDA یکسان نیست. برای مثال همین زبان ذکر شده در صورت سوال را نمیتوان توسط یک Single-Stack PDA نشان داد. در واقع میتوان گفت زبانهایی که Double-Stack PDA میپذیرد زیرمجموعه‌ای از زبانهایی است که Single-Stack PDA میپذیرد. برای این که این مورد را اثبات کنیم، کافیست نشان دهیم که چگونه با استفاده از یک Double-Stack PDA میتوانیم یک Single-Stack PDA را شبیه‌سازی کنیم. بدیهی است که برای این کار میتوانیم فقط از یکی از استک‌های PDA استفاده کنیم. در ادامه درس، با زبانهایی که میتوان آنها را با Double-Stack PDA نشان داد آشنا خواهید شد.

به نام خدا



نظريه زبانها و ماشينها - بهار ۱۴۰۲

تمرین شماره ۶

دستيary آموزشی اين مجموعه: پاشا براهييمى
pashabarahimi@gmail.com



تاریخ تحويل: ۶ اردیبهشت (صفحه درس)

نکته: در تمامی سوالاتی که باید PDA رسم کنید، فرض کنید \$ در استک قرار ندارد و در صورت نیاز باید خودتان این مورد را در استک push کنید.

(1) برای هرکدام از زبان‌های زیر یک PDA رسم کنید. (20 نمره)

(الف) $L_1 = \{a^i b^j c^k \mid j = 2i + k + 1; i, j, k \geq 0\}$

(ب) $L_2 = \{a^i b^j c^k \mid j \leq \max(i, k); i, j, k \geq 0\}$

(ج) $L_3 = \{a^n b^{3m} c^{2m} d^{n-1} \mid n, m \geq 1\}$

(د) $L_4 = \{xwy \mid x, y, w \in \{a, b\}^* \text{ and } w = w^R \text{ and } n_a(x) = n_b(y)\}$

(2) برای گرامر زیر یک PDA با حداقل 3 حالت^۱ رسم کنید و سپس زبان گرامر را بنویسید. (15 نمره)

$$S \rightarrow TbT$$

$$T \rightarrow aTb \mid bTa \mid TT \mid \epsilon$$

(3) زبان زیر را در نظر بگیرید: (20 نمره)

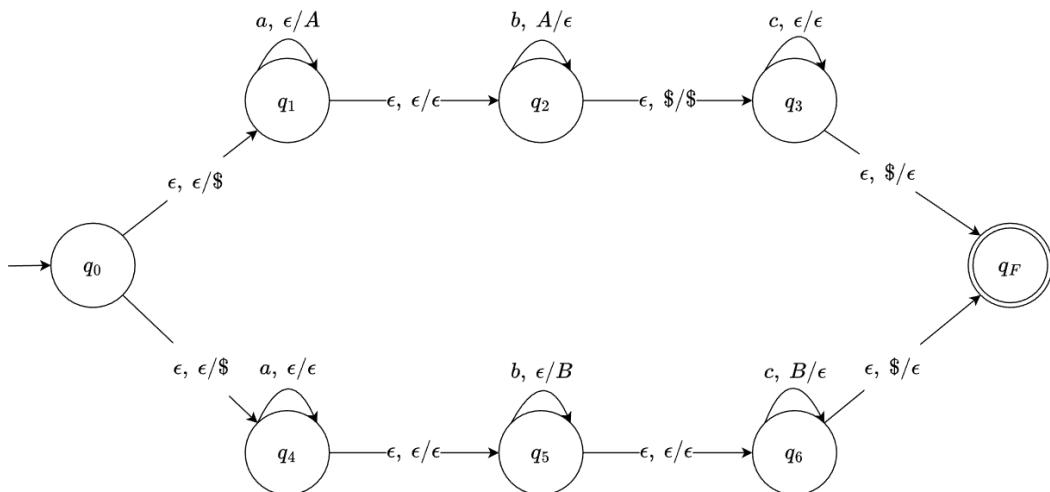
$$L = \{w \mid w = (ab)^* cc^* d\}$$

^۱ State

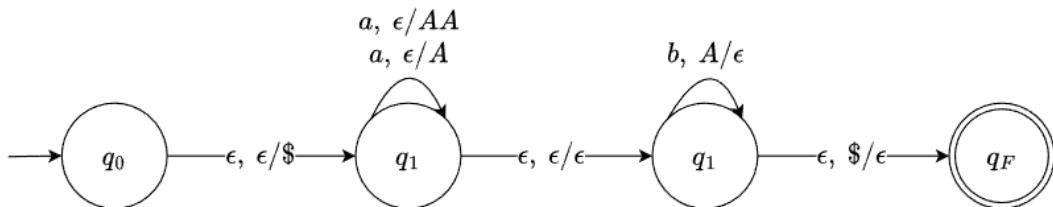
- الف) برای این زبان یک NFA رسم کنید.
- ب) آیا می‌توانید NFA بخش قبل را با فقط 3 حالت رسم کنید؟ در صورت امکان این کار را انجام دهید و در غیر این صورت، دلیل خود را به صورت خلاصه ذکر کنید.
- ج) آیا می‌توانید یک PDA برای این زبان با حداقل 3 حالت رسم کنید؟ در صورت امکان این کار را انجام دهید و در غیر این صورت، دلیل خود را به صورت خلاصه توضیح دهید.
- د) آیا می‌توان برای هر زبان Context-Free یک PDA با 3 حالت رسم کرد؟ در صورت امکان، روش کار خود را توضیح دهید.

(4) زبانی که هر کدام از PDA‌های زیر می‌پذیرد را بنویسید. (20 نمره)

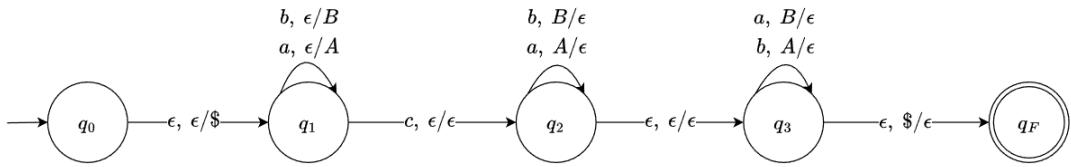
الف)



ب)



(5) گرامر متناظر با PDA زیر را بنویسید. (15 نمره)

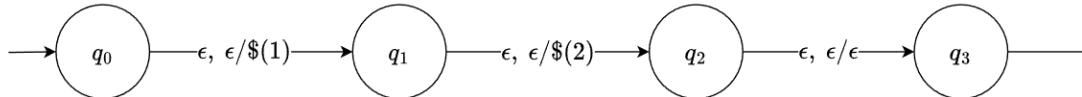


(6) آیا زبان زیر مستقل از متن است؟ پاسخ خود را اثبات کنید. (10 نمره)

$$L = \{a^n b^m \mid n \neq 3m + 1, n, m \geq 1\}$$

(7) امتیازی: به 2 استک PDA گفته می‌شود. در این Double-Stack PDA شما به 2 عدد استک دسترسی دارید و در هر گذار² می‌توانید فقط در یک استک عملیات انجام دهید و در هر عملیات مشخص می‌کنید که از کدام استک استفاده می‌کنید. (10 نمره)

برای رسم این PDA به مثال زیر توجه کنید:



در این مثال توسط گذار اول \\$ در استک اول push می‌شود و در گذار دوم \\$ در استک دوم push می‌شود. در گذار سوم به دلیل عدم انجام عملیات در استک، هیچ استکی مشخص نشده است.

الف) یک Double-Stack PDA برای زبان زیر رسم کنید.

$$L = \{a^n b^m c^m d^{n+m} \mid n, m \geq 0\}$$

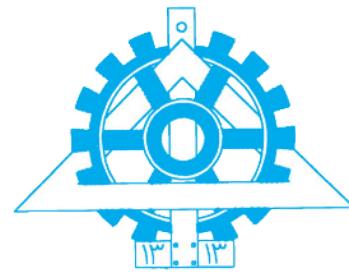
² Transition

ب) آیا قدرت Single-Stack PDA با Double-Stack PDA یکسان است؟ به صورت خلاصه دلیل خود را توضیح دهید. دقت کنید که برای برابر بودن قدرت این دو PDA، لازم است این برابری به صورت دو طرفه وجود داشته باشد؛ در نتیجه این مورد را در دلایل خود (در صورتی که پاسخ شما برابری قدرت است)، لاحظ کنید.



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



پاسخ تمرین شماره ۷
دستیار آموزشی این مجموعه: پاریاب مرادی
paryabmoradi1378@gmail.com

۱. با استفاده از لم تزریق (**Pumping lemma**), اثبات کنید که زبان مربوطه، عضو کلاس زبان‌های منظم نیست: (20 نمره)

a. $L = \{a^i b^j \mid j = i \text{ or } j = 2i\}$

با برهان خلف فرض کنید زبان منظم باشد و طول پمپ را p در نظر بگیرید. لم تزریق را برای رشته‌ی $s = a^p b^p$ اجرا می‌کنیم. طبق لم تزریق داریم

$$s = xyz, |xy| \leq p, |y| > 0 \Rightarrow y = a^j$$

اما

$$xy^2z = a^{p+j}b^p \notin L$$

تناقض حاصله نشان می‌دهد زبان مورد نظر منظم نیست.

b. $L = \{w \mid w \in \{a, b\}^* \text{ and } n_a(w) < 2n_b(w)\}$

فرض کنید زبان مورد نظر منظم و طول پمپ p است. لم تزریق را روی رشته $s = a^{2p-1}b^p$ اجرا کنید. مانند قسمت قبلی داریم

$$s = xyz, |y| > 0, |xy| \leq p \Rightarrow y = a^j$$

اما داریم

$$xy^2z = a^{p+j}b^p \notin L$$

این تناقض نشان می‌دهد زبان مورد نظر ما منظم نیست.

c. $L = \{a^{2^n} \mid n \geq 0\}$

فرض کنید این زبان منظم باشد و طول پمپ را p در نظر بگیرید. لم تزریق را برای رشته a^{2^p} اجرا می‌کنیم (دقت کنید طول این رشته از p بیشتر است). پس داریم

$$s = xyz, |y| > 0, |xy| \leq p \Rightarrow y = a^j, 1 \leq j \leq p$$

که نتیجه می‌دهد

$$xy^2z = a^{2^p+j} \in L$$

اما دقต کنید عدد $j + 2^p$ توانی از 2 نیست چون

$$2^p < 2^p + j < 2^p + p \leq 2^{p+1}$$

تناقض حاصله نشان می‌دهد L منظم نیست.

d. $L = \{w \in \{a, b\}^* \mid w = w_1bw_2b \dots bw_k, \text{ for } k \geq 0, \text{ each } w_i \in a^*, \text{ and } w_i \neq w_j \text{ for } i \neq j\}$

فرض کنید زبان مورد نظر منظم باشد و طول پمپ p باشد. لم تزریق را برای رشته $w = a^pba^{p+1}b \dots a^{2p-1}ba^{2p}$ اجرا می‌کنیم. مانند قسمت‌های قبل داریم

$$w = xyz, |y| > 0, |xy| \leq p \Rightarrow y = a^j, 1 \leq j \leq p$$

حال دقت کنید

$$xy^2z = a^{p+j}ba^{p+1}b \dots ba^{p+j}b \dots ba^{2p}$$

که عضوی از زبان L نیست چون دو بخش a^{p+j} دارد. تناقض حاصله نشان می‌دهد این زبان منظم نیست.

2. نسخه‌ی تغییریافته‌ای از لم تزریق را با تعریف زیر در نظر بگیرید:

اگر L یک زبان منظم باشد، ثابت n را می‌توان یافت، به نحوی که ازای هر z_1 و z_2 و z_3 و $z_1 z_2 z_3 \in L$

صدق کند و $|z_2| = n$ ، بتوانیم z_2 را به صورت uvw بازنویسی کنیم، طوری که $1 \geq |v| \geq |z_2|$ و به ازای هر

$i \geq 0$ رشتہ‌ی $z_1 u v^i w z_3$ عضو زبان L باشد. (15 نمره)

a. لم جدید را اثبات کنید.

n را برابر تعداد استیت‌های DFA این زبان در نظر می‌گیریم. فرض کنید پس از ورودی دادن رشتہ z_1

از استیت q_0 به استیت q از اتوماتا بررسیم و دنباله استیت‌های پس از ورودی دادن z_2 برابر

$q_n, q_{n-1}, \dots, q_1, q_0$ باشد. در انتها نیز با ورودی دادن z_3 از استیت q_n به استیت f می‌رود. طبق اصل

لانه کبوتری حداقل دوتا از این استیت‌ها مانند q_i و q_j یکی هستند و $j < i$. فرض کنید قسمت‌هایی از

رشته z_2 که برای قسمت‌های q_i, q_{i-1}, \dots, q_0 و q_j, q_{j-1}, \dots, q_0 هستند به ترتیب u و v و w

باشند. حال دقت کنید که رشتہ‌های $z_1 u$ و $w z_3$ اتومات را به ترتیب از استیت‌های q_i و q_j به

استیت‌های q_i و q_j می‌برند. با مطالب گفته شده به راحتی می‌توان دید که رشتہ $z_1 u v^i w z_3$ اتومات

را از استیت اولیه q به استیت نهایی f می‌برد که نشان می‌دهد عضوی از زبان L است.

b. با به کارگیری لم مربوطه، نشان دهید که $L = \{a^i b^j c^j \mid i, j \geq 1\}$ زبانی نامنظم است.

با برهان خلف فرض کنید این زبان منظم باشد و طول پمپ برابر p باشد. حال لم جدید را برای رشتہ

$(z_1 = a, z_2 = b^p, z_3 = c^p)$ اجرا می‌کنیم.

$$z_2 = uvw, |v| > 0 \Rightarrow v = b^j \Rightarrow z_1 u v^2 w z_3 = a^p b^{p+j} c^p$$

که به وضوح عضوی از زبان L نیست. تناقض حاصله نشان می‌دهد این زبان منظم نیست.

3. زبان‌های مستقل از متن قطعی نسبت به کدام یک از عملگرهای اجتماع (Union)، مکمل (Concatenation) و اتصال (Complementation) بسته هستند؟ اثبات کنید. (15 نمره)

در ابتدا اثبات می‌کنیم این زبان‌ها نسبت به مکمل بسته هستند. سپس با یک مثال نقض نشان می‌دهیم نسبت به اجتماع و اتصال بسته نیستند.

فرض کنید ماشین $DPDA = (Q, \Sigma, \Gamma, \delta, q_0, F)$ را داریم که زبان L را قبول می‌کند. در این صورت ماشین

مکمل زبان L را $DPDA' = (Q, \Sigma, \Gamma, \delta, q_0, F' = Q/F)$ قبول می‌کند. زیرا به دلیل قطعی بودن ماشین با

ورودی دادن رشته w ماشین به یک استیت مانند q می‌رسد. حال اگر $w \in L$ باشد داریم $w \in F$ و $w \notin L(DPDA')$ و اگر $w \notin L$ داریم $w \notin F$ و $w \in L(DPDA')$. این نشان می‌دهد که $L(DPDA') = L^c$ و زبان‌های مستقل از متن قطعی نسبت به مکمل بسته هستند.

این زبان‌ها نسبت به اجتماع بسته نیستند زیرا می‌دانیم زبان $\{a^n b^n | n \geq 0\} \cup \{a^n b^{2n} | n \geq 0\}$ مستقل از متن قطعی نیست ولی هر کدام از $\{a^n b^n | n \geq 0\}$ و $\{a^n b^{2n} | n \geq 0\}$ مستقل از متن قطعی هستند.

دو زبان مستقل از متن قطعی $c^* a^n b^n | n \geq 0$ را در نظر بگیرید. زبان اول چون منظم است، مستقل از متن قطعی نیز هست. زبان دوم نیز می‌توان با بررسی حرف اول رشته فهمید در کدام یک از زبان‌های مستقل از متن قطعی $\{a^n b^n | n \geq 0\}$ یا $\{a^n b^{2n} | n \geq 0\}$ باید ادامه داد که باعث می‌شود مستقل از متن قطعی باشد. حال نشان می‌دهیم زبان حاصل از اتصال این دو زبان مستقل از متن قطعی نیست. با برهان خلف فرض کنید $(c^* a^n b^n | n \geq 0) \cup (c^* a^n b^{2n} | n \geq 0)$ مستقل از متن قطعی باشد. اگر این زبان را با $c a^* b^*$ که زبانی منظم است اشتراک بگیریم به زبان مستقل از متن قطعی $\{c a^n b^n | n \geq 0\} \cup \{c a^n b^{2n} | n \geq 0\}$ می‌رسیم. چون تمامی رشته‌های این زبان با حرف c شروع می‌شوند می‌توان گفت پس از حذف آن c هم باز زبان مستقل از متن قطعی می‌ماند اما می‌دانیم $\{a^n b^n | n \geq 0\} \cup \{a^n b^{2n} | n \geq 0\}$ مستقل از متن قطعی نیست. تناقض حاصله نشان می‌دهد کلاس زبان‌های مستقل از متن قطعی نسبت به اتصال بسته نیست.

4. نشان دهید اگر L زبان مستقل از متن قطعی باشد و M یک زبان منظم باشد آنگاه زبان $L \cap M$ یک زبان مستقل از متن قطعی است. (10 نمره)

فرض کنید $DFA = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$ و $PDA = (Q_1, \Sigma, \Gamma, \delta_1, q_{01}, F_1)$ به ترتیب زبان‌های L و M را قبول کنند. در این صورت ماشین پشت‌های قطعی

$$PDA' = (Q' = Q_1 \times Q_2, \Sigma, \delta', q_0 = (q_{01}, q_{02}), F = F_1 \times F_2)$$

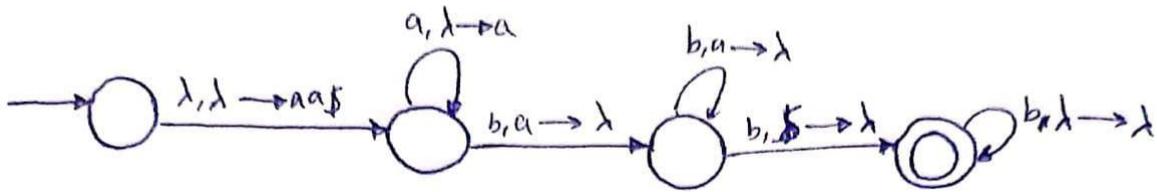
که δ' به صورت زیر تعریف شده است زبان $L \cap M$ را قبول می‌کند.

$$\forall q' = (q'_1, q'_2) \in Q', c \in \Sigma, \gamma \in \Gamma, \delta_1(q'_1, c, \gamma) = (q'^c_1, \gamma'): \delta'(q', c, \gamma) = ((q'^c_1, \delta_2(q'_2, c)), \gamma')$$

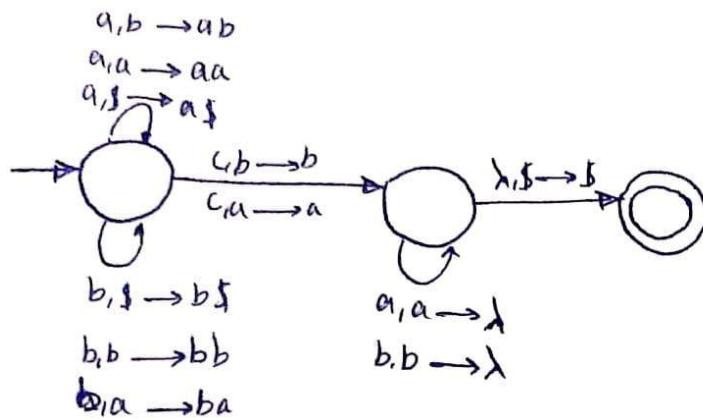
به صورت شهودی با این تعریف 'delta' ماشین سازی اجرای رشته روی DFA و PDA را همزمان انجام می‌دهیم و حالت هر دو ماشین را با یک زوج مرتب نشان می‌دهیم. حال واضح است که رشته در هر دو زبان وجود دارد اگر استیت نهایی عضوی از $F_1 \times F_2$ باشد. پس ماشین ارائه شده زبان $L \cap M$ را می‌پذیرد.

5. برای زبان‌های زیر DPDA رسم کنید. ($\Sigma = \{a, b\}$) (20 نمره)

a. $L = \{a^n b^m \mid m \geq n + 3\}$



b. $L = \{wcw^R \mid w \in \{a, b\}^*\}$



6. در خصوص ماشین‌های پشته‌ای دارای قطعیت (DPDA) به پرسش‌های زیر پاسخ دهید: (20 نمره)

- a. نشان دهید که برای هر زبان منظم، یک ماشین پشته‌ای قطعی قابل رسم است که رشتة‌های آن زبان را می‌پذیرد، تنها دو حالت دارد، هیچ گذار ع ندارد و در آن نمادها هیچ گاه از پشته حذف نمی‌شوند.

از آنجا که زبان‌های منظم زیرمجموعه‌ی محض زبان‌های مستقل از متن هستند، برای همه‌ی این زبان‌ها می‌توانیم یک ماشین پشته‌ای طراحی کنیم. در این سوال، هدف این است که نشان دهیم چگونه می‌توانیم عملکرد یک ماشین متنه‌ای را به کمک یک ماشین پشته‌ای شبیه‌سازی کنیم.

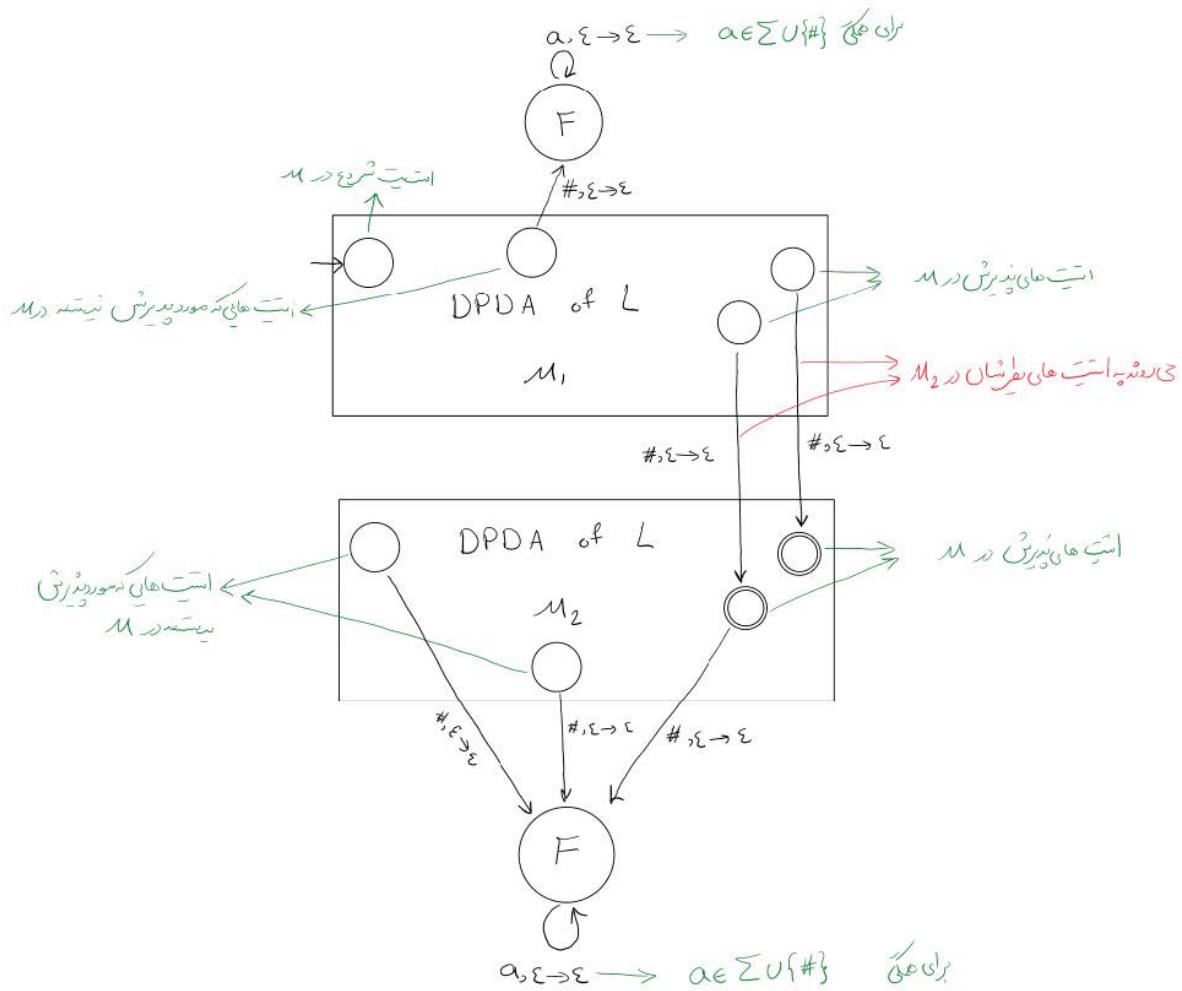
می‌دانیم که در ماشین‌های متنه‌ای، صرفاً داشتن نماد ورودی که هد خوانش روی آن قرار گرفته، و نیز حالت فعلی ماشین، برای تعیین حالت بعدی کافیت می‌کند. به همین جهت کافی است که از پشته، به همین منظور استفاده کنیم. یک روش آن است که به ازای هر حالت ماشین متنه‌ای، یک نماد در نظر بگیریم. هر بار که ماشین متنه‌ای به یک حالت جدید تغییر حالت می‌دهد، ماشین پشته‌ای که قرار است عملکرد آن را شبیه‌سازی کند، نماد مربوطه را بالای پشته push می‌کند. حال برای تعیین حالت بعدی، به حالت

قبلی که نماد آن در بالای پشته قرار گرفته دسترسی دارد و به همین جهت، می‌تواند برای نماد بعدی تصمیم‌گیری کند.

ماشین پشته‌ای مربوطه، بر حسب اینکه شیوه‌ی پذیرش در آن چگونه باشد، می‌تواند حداکثر با دو یا سه حالت پیاده‌سازی گردد.

b. نشان دهید که اگر برای L یک ماشین پشته‌ای قطعی وجود داشته باشد که رشته‌های این زبان را بپذیرد، یک ماشین پشته‌ای قطعی دیگر قابل تعریف است که رشته‌های زبان $\{ x \# y \mid x \in L, xy \in L \}$ را بپذیرد. فرض می‌کنیم نماد $\#$ در هیچ‌یک از رشته‌های L ظاهر نمی‌شوند.

مطابق شکل پایین اگر داشتم باشیم که M یک DPDA باشد که $L(M) = L$ آنگاه داریم که ما با استفاده از دو تا کپی از M با نام‌های M_1 و M_2 می‌توانیم که زبان خواسته شده را تولید کنیم به این صورت که هرگاه در ماشین M_1 هستیم و ورودی رشته ما $\#$ نباشد همان روند عادی M را ادامه می‌دهیم اما هر وقت $\#$ بباید دو حالت دارد، حالت اول این است که ما در یک استیت پذیرش در M قرار داریم (در اصل آن استیتی که ما در M_1 قرار داریم خودش در ماشینی که تولید می‌کنیم پذیرش نیست اما استیت نظریش در M پذیرش است) در این صورت ما به استیت نظری آن در M_2 می‌رویم و ادامه می‌دهیم، در غیر این صورت به یک استیت F می‌رویم که در اصل استیتی است که نشان می‌دهد ما دیگر نمی‌توانیم به یک حالت پذیرش برسیم و از آن استیت دیگر خارج نمی‌شویم. حال اگر در M_2 باشیم نیز اگر رشته ورودی ما $\#$ نباشد روال عادی M را طی می‌کنیم تا به یک استیت پذیرش برسیم. در هنگام پایان شدن رشته (استیت‌های پذیرش در M_2 همان استیت‌های پذیرش در M است) اما اگر رشته ورودی ما $\#$ باشد در هر حالتی به F می‌رویم. توضیحاتی مختصر برای درک بیشتر در شکل آمده است.



7. (امتیازی) قضیه‌ی زیر معروف به لم اوگدن را در نظر گرفته، با استفاده از آن اثبات کنید که موارد a و b زبان‌های مستقل از متن نیستند. (20 نمره)

Ogden's Lemma: فرض کنید که L یک زبان مستقل از متن باشد. ثابت n وجود دارد، به نحوی که اگر w یک رشته‌ی دلخواه عضو زبان L باشد، و ما n تا، یا بیشتر، از نمادهای w را انتخاب کنیم، بتوانیم w را به صورت $uvxyz$ بنویسیم، به نحوی که سه شرط مقابل ارضاء شوند:

1. u و y روی هم دستکم یک نماد منتخب داشته باشند

2. vxy حداکثر n نماد منتخب داشته باشد

3. به ازای هر $i \geq 0$ رشته‌ی $uv^i xy^i z$ عضو زبان L باشد.

$$\text{a. } L = \{ a^p b^q c^r d^s \mid p = 0 \text{ or } q = r = s \}$$

فرض کنید این زبان مستقل از متن باشد و طول پمپ را n بنامید. حال لم اوگدن را روی رشته

$w = ab^n c^n d^n$ اجرا می‌کنیم. فرض کنید تمام حروف a, b, c, d علامت‌گذاری شده باشند. در این

صورت داریم $w = uvxyz$ که تعداد حروف علامتدار vxy حداکثر n است و uv حداقل یک حرف علامتدار دارد. در این صورت مانند تمام مسائل لم تزریق به راحتی می‌توان ثابت کرد که v و y تنها شامل یک حرف هستند چون اگر حداقل دو حرف متفاوت داشته باشد فرم $a^* b^* c^* d^*$ در uv^2xy^2z از بین می‌رود. حال دقت کنید که تعداد حداکثر دو تا از حرف‌های c, b, d , یکی می‌ماند که تناقض است. پس این زبان مستقل از متن نیست.

$$\text{b. } L = \{ a^n b^n c^i \mid i \neq n \}$$

فرض کنید این زبان مستقل از متن باشد و طول پمپ را n در نظر بگیرید. حال لم اوگدن را روی رشته $w = a^{n!+n} b^{n!+n} c^n$ اجرا می‌کنیم که تمام حروف c علامتگذاری شده‌اند. پس داریم $w = uvxyz$ که حداکثر n حرف c دارد و vy نیز شامل حداقل یک حرف c هست و برای هر $i \geq 0$ رشته $uv^i xy^i z$ نیز در زبان هست. دقیقاً مانند قسمت قبلی می‌توان گفت که رشته‌های v و y شامل حداکثر یک حرف متفاوت هستند. در این صورت دو حالت ممکن برای آن‌ها وجود دارد:

$$\text{حالت اول: } v = a^s \text{ or } b^s, \quad y = c^k$$

در این حالت اگر به رشته uv^2xy^2z دقت کنید متوجه می‌شوید که تعداد a و b های آن متفاوت است چون نسبت به $uvxyz$ تعداد تنها یکی از این دو حرف تغییر کرده است. پس این رشته در L نیست که تناقض است.

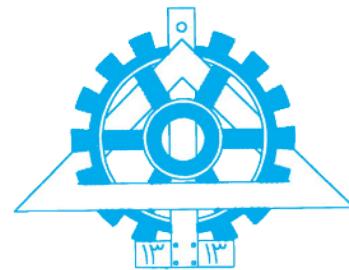
$$\text{حالت دوم: } v = c^s, \quad y = c^k$$

در این صورت با انتخاب $i = \frac{n!}{s+k}$ داریم $uv^i xy^i z = a^{n!+n} b^{n!+n} c^{n!+n} \in L$ که تناقض است. دقت کنید که انتخاب $i = \frac{n!}{s+k}$ قابل انجام است زیرا طبق فرض $n \leq |vxy|$ که نشان می‌دهد $n \leq s + k$ و تضمین می‌کند که i عددی طبیعی است.



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۷

دستیار آموزشی این مجموعه: پاریاب مرادی

paryabmoradi1378@gmail.com

تاریخ تحويل: ۱۲/۰۲/۱۴۰۲ ۰۵:۵۹

۱. با استفاده از لم تزریق (Pumping lemma)، اثبات کنید که زبان مربوطه، عضو کلاس زبان‌های منظم

(20 نمره)

- a. $L = \{a^i b^j \mid j = i \text{ or } j = 2i\}$
- b. $L = \{w \mid w \in \{a, b\}^* \text{ and } n_a(w) < 2n_b(w)\}$
- c. $L = \{a^{2^n} \mid n \geq 0\}$
- d. $L = \{w \in \{a, b\}^* \mid w = w_1 bw_2 b \dots bw_k, \text{ for } k \geq 0, \text{ each } w_i \in a^*,$
 $\text{and } w_i \neq w_j \text{ for } i \neq j\}$

۲. نسخه‌ی تغییریافته‌ای از لم تزریق را با تعریف زیر در نظر بگیرید:

اگر L یک زبان منظم باشد، ثابت n را می‌توان یافت، به نحوی که به ازای هر $z_1, z_2, z_3 \in L$ و z_3 که

صدق کند و $n = |z_2|$ ، بتوانیم z_2 را به صورت uvw بازنویسی کنیم، طوری که $1 \leq |v| \leq n$ و به ازای هر

$i \geq 0$ رشتی $z_1 uv^i w z_3$ عضو زبان L باشد. (15 نمره)

a. لم جدید را اثبات کنید.

b. با به کارگیری لم مربوطه، نشان دهید که $L = \{a^i b^j c^j \mid i, j \geq 1\}$ زبانی نامنظم است.

۳. زبان‌های مستقل از متن قطعی نسبت به کدام یک از عملگرهای اجتماع (Union)، مکمل (Complement) و اتصال (Concatenation) بسته هستند؟ اثبات کنید. (15 نمره)

۴. نشان دهید اگر L زبان مستقل از متن قطعی باشد و M یک زبان منظم باشد آنگاه زبان $L \cap M$ یک زبان مستقل از متن قطعی است. (10 نمره)

۵. برای زبان‌های زیر DPDA رسم کنید. ($\Sigma = \{a, b\}\}$) (20 نمره)

- a. $L = \{a^n b^m \mid m \geq n + 3\}$
- b. $L = \{wcw^R \mid w \in \{a, b\}^*\}$

6. در خصوص ماشین‌های پشته‌ای دارای قطعیت (DPDA) به پرسش‌های زیر پاسخ دهید: (20 نمره)
- نstan دهید که برای هر زبان منظم، یک ماشین پشته‌ای قطعی قابل رسم است که رشتة‌های آن زبان را می‌پذیرد، تنها دو حالت دارد، هیچ گذار ع ندارد و در آن نمادها هیچ‌گاه از پشته حذف نمی‌شوند.
 - نstan دهید که اگر برای L یک ماشین پشته‌ای قطعی وجود داشته باشد که رشتة‌های این زبان را بپذیرد، یک ماشین پشته‌ای قطعی دیگر قابل تعریف است که رشتة‌های زبان $\{x \in L, xy \in L \mid x \# y\}$ را بپذیرد. فرض می‌کنیم نماد $\#$ در هیچ‌یک از رشتة‌های L ظاهر نمی‌شوند.
7. (امتیازی) قضیه‌ی زیر معروف به لم اوگن را در نظر گرفته، با استفاده از آن اثبات کنید که موارد a و b زبان‌های مستقل از متن نیستند. (20 نمره)

a. **Ogden's Lemma**: فرض کنید که L یک زبان مستقل از متن باشد. ثابت n وجود دارد، به نحوی که اگر w یک رشتة‌ی دلخواه عضو زبان L باشد، و ما n نا، یا بیشتر، از نمادهای w را انتخاب کنیم، بنوایم w را به صورت $uvxyz$ بنویسیم، به نحوی که سه شرط مقابل ارضاء شوند:

1. v و y روی هم دستکم یک نماد منتخب داشته باشند

2. uxy حداقل n نماد منتخب داشته باشد

3. به ازای هر $i \geq 0$ رشتة‌ی $uv^i xy^i z$ عضو زبان L باشد.

$$a. \quad L = \{a^p b^q c^r d^s \mid p = 0 \text{ or } q = r = s\}$$

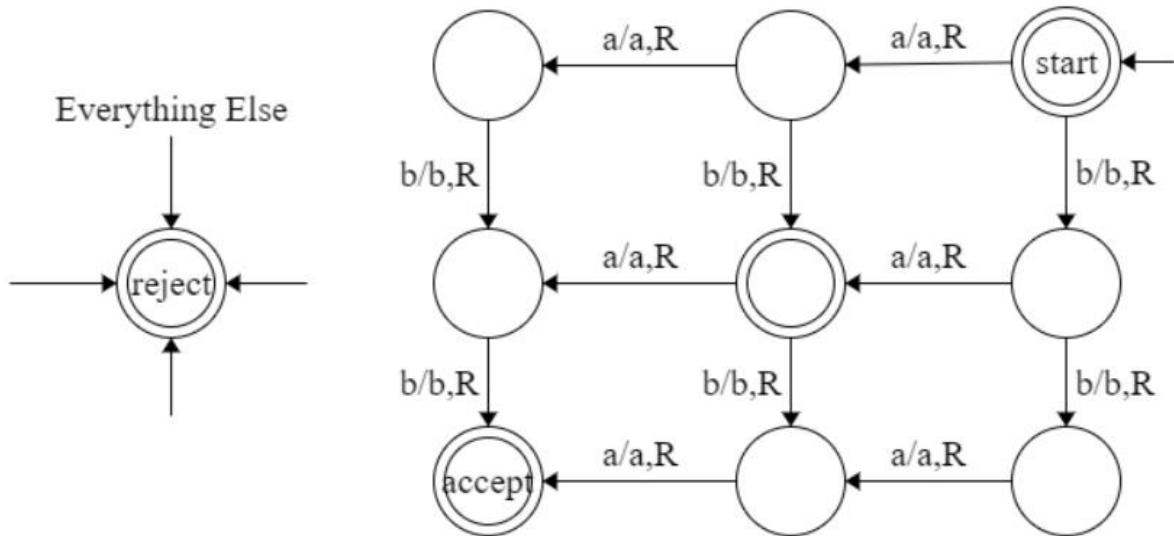
$$b. \quad L = \{a^n b^n c^i \mid i \neq n\}$$



۱. برای هریک از زبان‌های زیر ماشین تورینگ متناظر را توصیف کنید. (منظور از توصیف ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به جابجایی head می‌باشد)

$$\text{الف) } L = \{w \mid N_a(w) \equiv N_b(w) \pmod{3}, w \in (a+b)^*\}$$

پاسخ: در این ماشین باقیمانده تقسیم تعداد a و b به ۳ باید یکسان باشد.

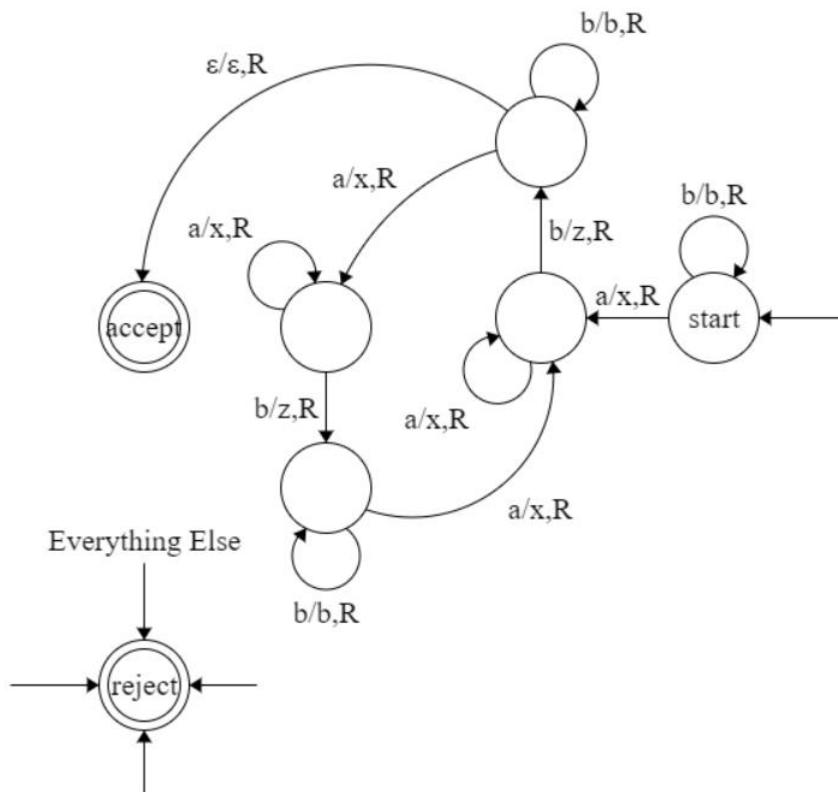


$$\text{ب) } L = \{w \mid N_a(w) = 3N_b(w), w \in (a+b)^*\}$$

در این ماشین باید به ازای هر b , a تا ۳ علاممت بخورد. از اول نوار شروع می‌کنیم و با هر بار دیدن یک b , آنرا علامت زده و به اول نوار (یا آخرین a خط خورده) برمی‌گردیم. بعد به دنبال ۳ عدد a می‌گردیم و در صورت یافتن آنها دوباره به اول نوار (یا آخرین b خط خورده) بر می‌گردیم. این کار را آنقدر ادامه می‌دهیم تا b ‌ها تمام شوند. در صورتی که به ازای b دیگر a نمانده بود باید reject کنیم.

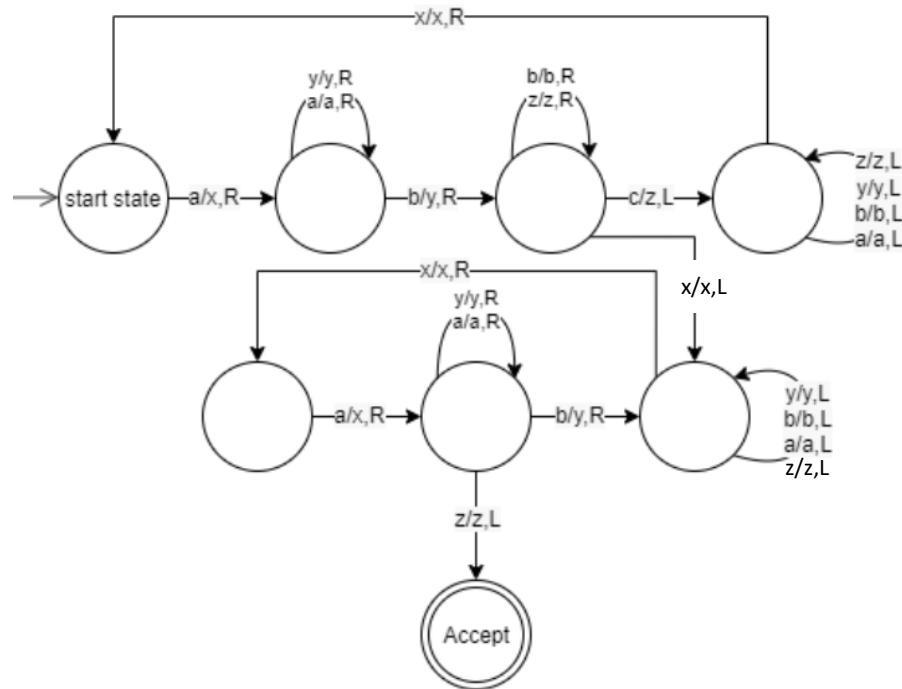
$$L = \{w \mid N_{ab}(w) \equiv 1 \pmod{2}, w \in (a+b)^*\}$$

پاسخ: در این مسئله قصد داریم ماشین تورینگی طراحی کنیم که زبانی را قبول کند که رشته‌ها با فرد عدد زیر رشته 'ab' را قبول می‌کند. اول باید یک a پیدا کنیم و آنرا علامت X بزنیم، اگر بعد آن b داشتیم باید یک علامت مناسب (برای مثال Z) رو آن بزنیم و در غیر این صورت (که a دیدیم یا به پایان رسیدیم) باید یا دوباره X بزنیم یا بر اساس فرد یا زوج بودن تعداد ab قبول یا رد کنیم:



۲. توضیح دهید هر کدام از ماشین تورینگ‌های زیر چه زبانی را قبول می‌کنند یا چه تغییراتی روی رشته ورودی اعمال می‌کنند.

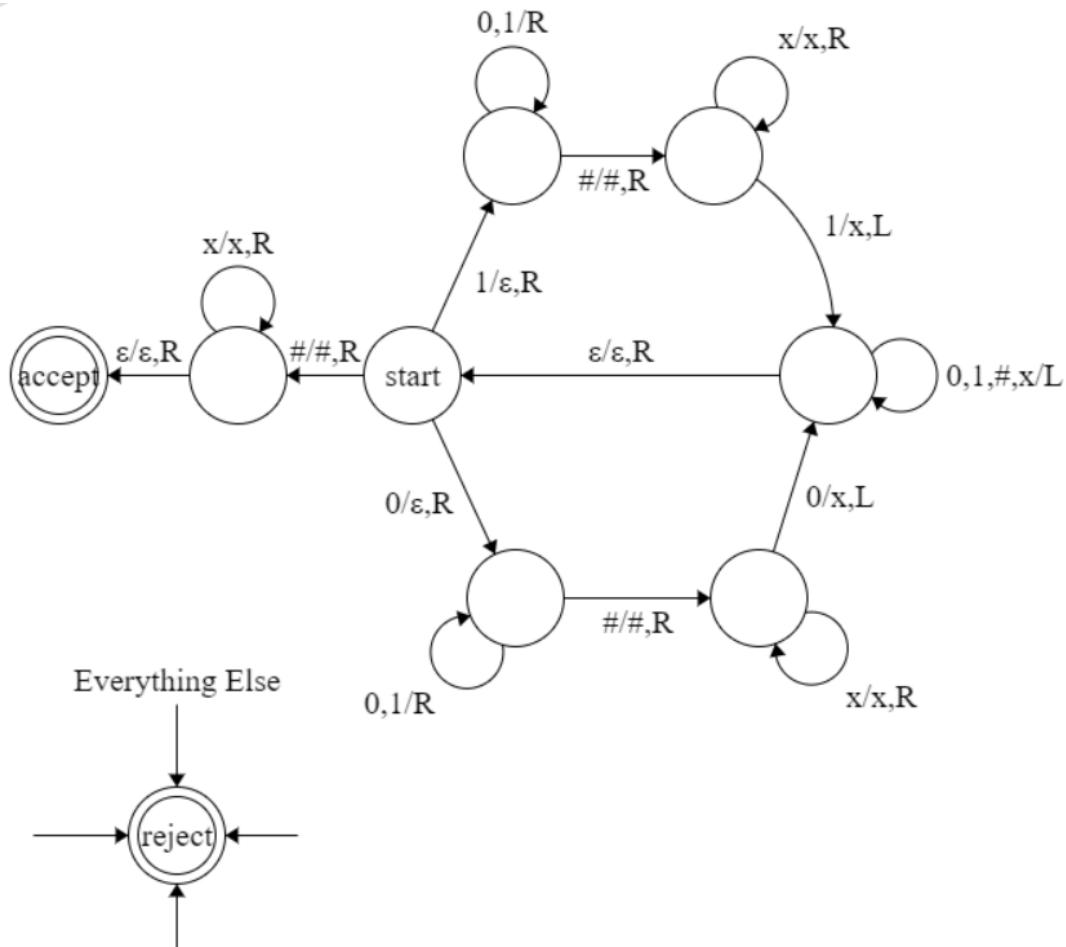
(الف)



پاسخ: در این مسئله در هر گام یکی از a ها و یکی از b ها و یکی از c ها را خط میزنیم و اگر c تمام شد برای a و b این کار را ادامه داده تا در نهایت اگر تنها a وجود داشت یا رشته تمام شده بود، قبول می‌کنیم. این ماشین زبان زیر را قبول می‌کند:

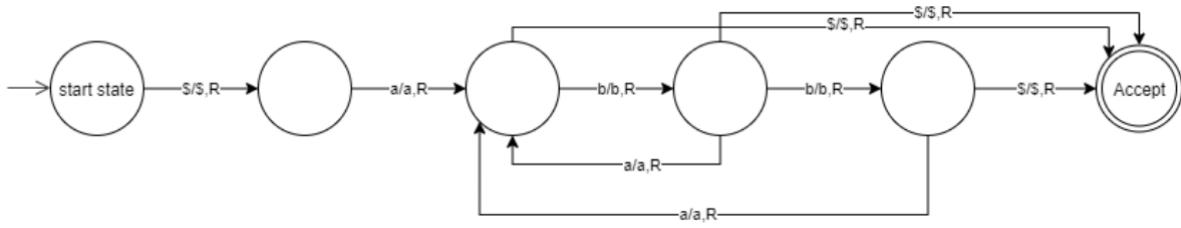
$$L = \{a^i b^j c^k \mid i \geq j \geq k, k \geq 1\}$$

ب) نکته: به $\#$ و فرمت ورودی قابل پذیرش دقیق کنید.



پاسخ: این ماشین رشته‌های متقارن (palindrome) را قبول می‌کند:
 $L = \{w\#w \mid w \in (0 + 1)^*\}$

(ج)



پاسخ: این ماشین رشته‌هایی که متشکل از زیررشته‌های 'ab'، یا 'abb' هستند را قبول می‌کند. اول و آخر نوار هم با \\$ مشخص شده است.

$$L = \{x_1x_2\dots x_n \mid n \geq 0, \forall i > 0 : x_i = ab \vee x_i = abb\}$$

۳. ماشین تورینگی طراحی کنید که خروجی توابع زیر را روی نوار قرار دهد. دقیق کنید در این سوال تمامی ورودی‌های توابع (x و y) بصورت unary هستند. (منظور از طراحی ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به جابجایی head می‌باشد)

$$\text{الف) } f(x) = x + 1$$

پاسخ: در این مسئله صرفاً به انتهای نوار می‌رویم و یک 1 به انتهای نوار اضافه می‌کنیم.

$$\text{ب) } f(x, y) = x - y$$

پاسخ: فرض کنیم ورودی‌ها با یک # از هم جدا شده اند (11#11). سمت راست # ورودی x و سمت چپ ورودی y خواهد بود. از اول نوار شروع می‌کنیم و به ازای هر 1 سمت راست یک 1 از سمت چپ خط میزیم. در نهایت اگر سمت راست # 1 مانده بود پاسخ 1‌های سمت راست است و اگر سمت چپ مانده بود پاسخ 1‌های سمت چپ خواهد بود که یک عدد منفی است.

$$\text{ج) } f(x, y) = xy$$

پاسخ: در این بخش به تعداد y بار باید x روی نوار تکرار شود. ورودی‌ها با # جدا شده اند و انتهای نوار یک x قرار میدهیم که سمت راست آن پاسخ خواهد بود. از اول نوار شروع می‌کنیم تا به # برسیم بعد اولین 1 را علامت میزیم و به اول نوار برミگردیم. حال به ازای هر 1 که اول نوار هست باید به انتهای نوار 1 اضافه کنیم که این کار با علامت زدن 1 اول نوار، حرکت به انتهای اضافه کردن 1، بازگشت و علامت زدن 1‌های بعدی تا رسیدن به # میسر می‌شود. بعد هم برミگردیم و دوباره علامت 1‌ها را برミداریم. این کار تا علامت خوردن تمامی 1‌های بعد # و قبل x تکرار می‌شود.

۴. ماشین تورینگی توصیف کنید که زبان زیر را قبول کند:

$$L = \{a^n b^{m+1} | n, m > 0, m \leq n\}$$

الف) تعریف رسمی^۱ ماشین تورینگ را بنویسید.

ب) state diagram ماشین را رسم کنید.

ج) نوار ماشین را به ازای ورودی‌های aaaa و aabb رسم کنید.

پاسخ:

$$Q = \{q_1, q_2, q_3, q_4, q_5, \text{reject}\}$$

$$\Sigma = \{a, b\}$$

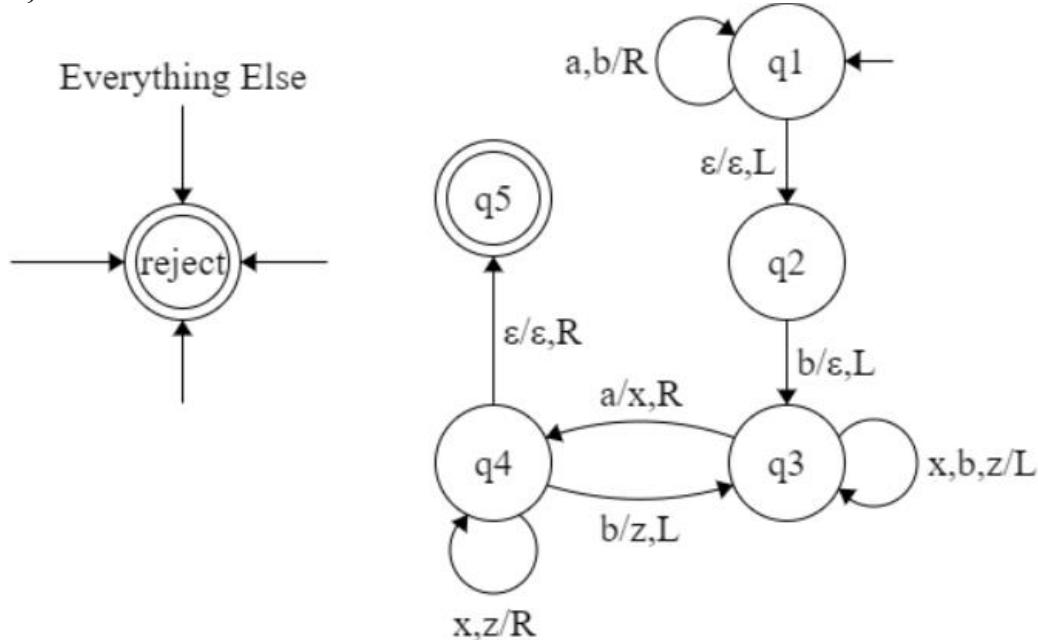
$$\Gamma = \{a, b, x, z, \varepsilon\}$$

$$\text{init} = q_1$$

$$q_{\text{acc}} = q_5$$

$$q_{\text{reject}} = \text{reject}$$

$$\delta = \{(q_1, a) \rightarrow (q_1, a, R), (q_1, b) \rightarrow (q_1, b, R), (q_1, \varepsilon) \rightarrow (q_2, \varepsilon, L), (q_2, b) \rightarrow (q_3, \varepsilon, L), \dots\}$$



به ازای ورودی aaaa ماشین accept می‌کند اما با aabb ماشین reject می‌کند:

a	a	a	a	
a	a	a	a	
a	a	a	a	
a	a	a	a	
a	a	a	a	

¹ Formal Definition

a	a	b	b	
a	a	b	b	
a	a	b	b	
a	a	b	b	
a	a	b	b	
a	a	b	b->ε	
a	a	b	ε	
a	a->x	b	ε	
a	x	b->z	ε	
a	x	z	ε	
a->x	x	z	ε	
...				
x	x	z	ε	

۵. ماشین تورینگی طراحی کنید که تابع زیر را پیاده سازی کند. ارائه شبه کد به همراه توضیحات لازم کافی است.

$$f(x) = 3^x + x^3$$

پاسخ: شبه کد تابع را برای مشخص کردن الگوریتم مینویسیم:

```
func f(x):
    cube <- x
    cube <- cube *x
    cube <- cube *x
    exp <- 1
    i <- 1
    while i <= x do
        exp <- exp*3
    endwhile
    return exp+cube
endfunc
```

۶ ماشین تورینگی طراحی کنید که رشته های ab را دو برابر می کند. برای مثال با وارد کردن عبارت $aaba$ در نهایت خروجی $aababa$ می شود. (منظور از طراحی ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به جابجایی head می باشد)

پاسخ: عملکرد ماشین به صورت زیر است:

نخست اولین زیررشته‌ای که دوباره نشده را پیدا کن. سپس آنرا علامت بزن. بعد به انتهای ورودی برس.

همه چیز را از انتهای ورودی به آخرین کاراکتر علامت‌گذاری شده به چپ دوبار شیفت بده تا جا برای کپی زیررشته روی نوار باز شود.

سپس با نوشتن ab روی جاهای خالی، زیررشته را کپی کنه و بعد به سمت راست برو. در اینجا لازم است تا ab جدید هم علامت‌گذاری شود تا دوباره کپی نشود.

این کارها را تکرار کن تا در نهایت تمامی زیررشته‌های ab ورودی را علامت‌گذاری کرده باشی (در واقع تمامی ab ها را علامت‌گذاری کرده باشی) بعد هم علامتها را بردار.

۷. (امتیازی) رمزنگاری جانشینی^۲ یکی از روش‌های رمزنگاری پیام‌های مخابره شده بوده. این روش امنیت بالایی ندارد و با حملات ساده قابل شکست است. در این مدل رمزنگاری به ازای هر حرف رشته ورودی حرف دیگری جایگزین می شود. برای مثال رشته flat با عبور از این رمزنگار می‌تواند تبدیل به ghzy شود. برای اطلاعات بیشتر از این مدل رمزنگاری می‌توانید به اینترنت مراجعه کنید. توجه داریم که این رمزنگاری قاعده خاصی برای تخصیص هر کاراکتر ندارد. ما مسلط به زبان مبدأ دستگاه هستیم و بی نهایت رشته به آن زبان داریم. ماشین تورینگی طراحی کنید که رشته رمزنگاری شده را بازگشایی کند. (توصیف سطح بالا از ماشین کافی است)

پاسخ: اگر تمامی جایگشت های ممکن برای تناظر یک حرف به حرف دیگری را بسازیم (که برای یک زبان n حرفی $n!$ حالت خواهد بود) می‌توان با کمک لغتنامه ای که در اختیار داریم پیام را دیکد کرد. پس به همین صورت عمل می‌کنیم. دو راه داریم: اول تمامی حالات ممکن را برای ورودی تولید کنیم، بعد هر کدام را با لغت نامه در دسترس مقایسه کنیم. به این صورت که کلمات تولید شده را با کلمات لغت نامه مقایسه کرد و آنها یکی که مچ شدند را مشخص می‌کنیم. اگر تمامی کلمات به مفهوم مناسبی رسیده بودند حالات مورد بررسی قابل قبول خواهد بود. می‌توان تک تک حالات را هم اول تولید کرد بعد مقایسه و بعد به سراغ حالت بعدی برویم.

² Substitution cipher



تاریخ تحویل: ۲۰ اردیبهشت(صفحه درس)

- برای هریک از زبان‌های زیر ماشین تورینگ متناظر را توصیف کنید. (منظور از توصیف ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به جابجایی head می‌باشد)

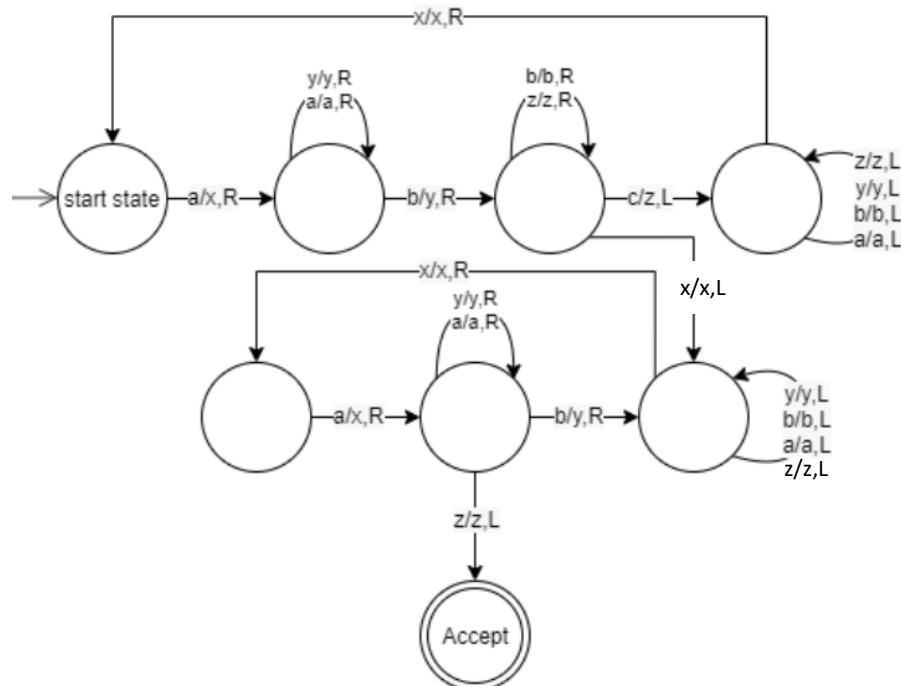
(الف) $L = \{w \mid N_a(w) \equiv N_b(w) \pmod{3}, w \in (a+b)^*\}$

(ب) $L = \{w \mid N_a(w) = 3N_b(w), w \in (a+b)^*\}$

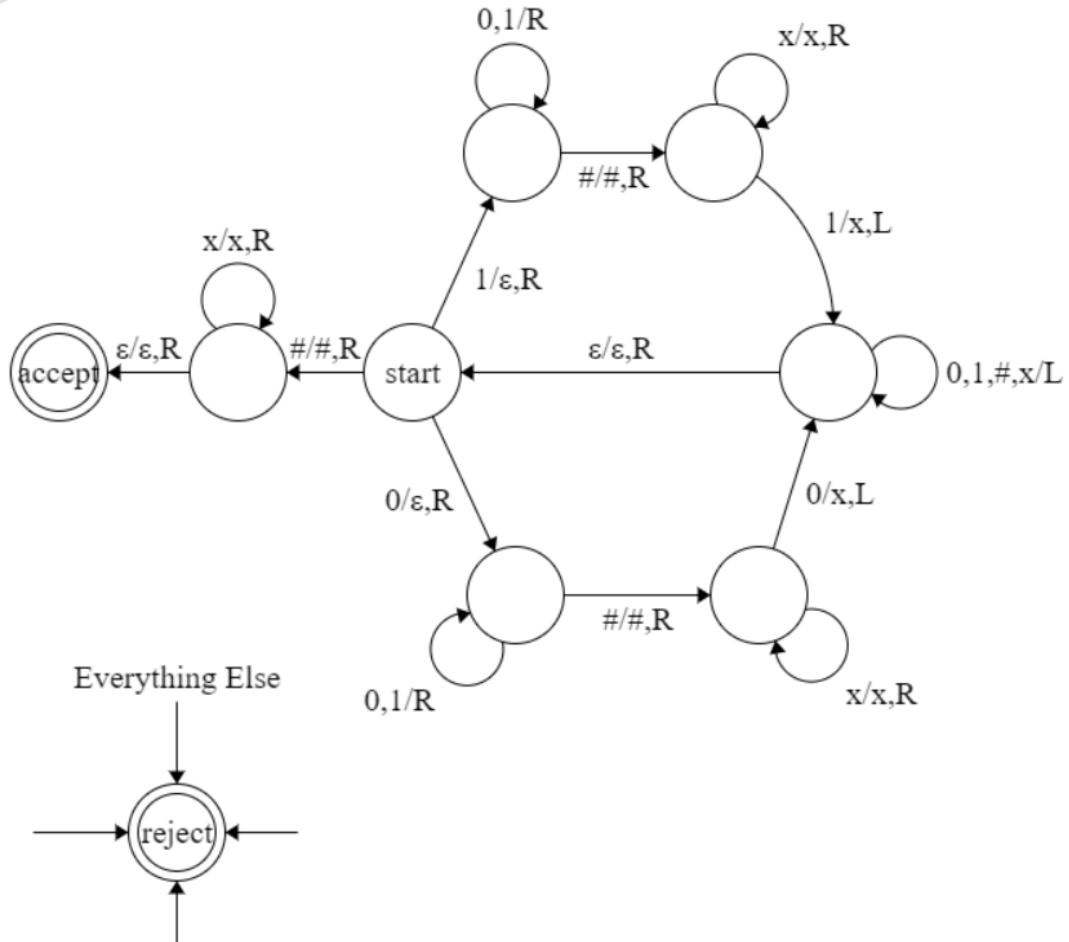
(ج) $L = \{w \mid N_{ab}(w) \equiv 1 \pmod{2}, w \in (a+b)^*\}$

- توضیح دهید هر کدام از ماشین تورینگ‌های زیر چه زبانی را قبول می‌کنند یا چه تغییراتی روی رشته ورودی اعمال می‌کنند.

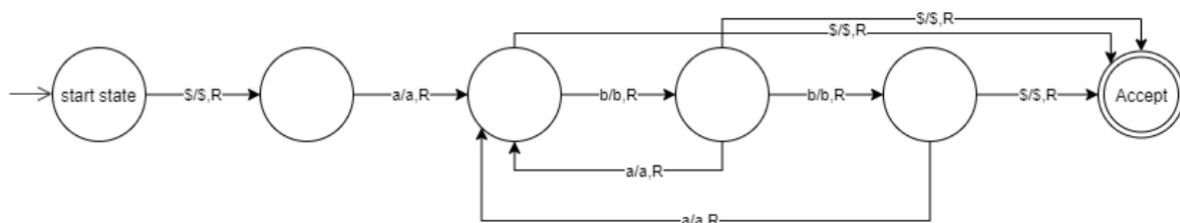
(الف)



ب) نکته: به # و فرمت ورودی قابل پذیرش دقت کنید.



(ج)



۳. ماشین تورینگی طراحی کنید که خروجی توابع زیر را روی نوار قرار دهد. دقت کنید در این سوال تمامی ورودی‌های توابع (x و y) بصورت unary هستند. (منظور از طراحی ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به حابچایی head می‌باشد)

$$\text{الف) } f(x) = x + 1$$

$$\text{ب) } f(x, y) = x - y$$

$$\text{ج) } f(x, y) = xy$$

۴. ماشین تورینگی توصیف کنید که زبان زیر را قبول کند:

$$L = \{a^n b^{m+1} | n, m > 0, m \leq n\}$$

الف) تعریف رسمی^۱ ماشین تورینگ را بنویسید.

ب) state diagram ماشین را رسم کنید.

ج) نوار ماشین را به ازای ورودی‌های aaaa و aabb رسم کنید.

۵. ماشین تورینگی طراحی کنید کهتابع زیر را پیاده سازی کند. ارائه شبکه کد به همراه توضیحات لازم کافی است.

$$f(x) = 3^x + x^3$$

۶. ماشین تورینگی طراحی کنید که رشته های ab را دو برابر می‌کند. برای مثال با وارد کردن عبارت aaba در نهایت خروجی aababa می‌شود. (منظور از طراحی ماشین تورینگ طراحی به همراه state diagram و تمامی جزئیات مربوط به جابجایی head می‌باشد)

۷. (امتیازی) رمزنگاری جانشینی^۲ یکی از روش‌های رمزنگاری پیام‌های مخابره شده بوده. این روش امنیت بالایی ندارد و با حملات ساده قابل شکست است. در این مدل رمزنگاری به ازای هر حرف رشته ورودی حرف دیگری جایگزین می‌شود. برای مثال رشته flat با عبور از این رمزنگار می‌تواند تبدیل به ghzy شود. برای اطلاعات بیشتر از این مدل رمزنگاری می‌توانید به اینترنت مراجعه کنید. توجه داریم که این رمزنگاری قاعده خاصی برای تخصیص هر کاراکتر ندارد. ما مسلط به زبان مبدأ دستگاه هستیم و بی نهایت رشته به آن زبان داریم. ماشین تورینگی طراحی کنید که رشته رمزنگاری شده را بازگشایی کند. (توصیف سطح بالا از ماشین کافی است)

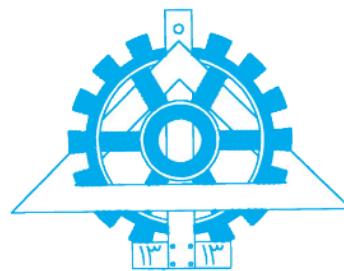
¹ Formal Definition

² Substitution cipher



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۲



پاسخ تمرین شماره ۹
دستیار آموزشی این مجموعه: سپهر آزردار

sepehr81sepehr@gmail.com

۱. در ماشین تورینگ استاندارد، اگر در ابتدای نوار باشیم و بخواهیم به سمت چپ حرکت کنیم، آنگاه در همان جا خواهیم ماند. ماشین تورینگ M' را در نظر بگیرید. این ماشین در شرایط گفته شده به ترپ استیت می‌رود و در همان جا برای همیشه خواهد ماند. نشان بدھید که چگونه میتوان ماشین تورینگ استاندارد را به ماشین تورینگ گفته شده تبدیل کرد به طوری که هر دو یک زبان را بپذیرند.

پاسخ:

ماشین تورینگ M را در نظر بگیرید، میخواهیم M را به ماشین تورینگ M' تبدیل کنیم به نحوی که هر دو یک زبان را بپذیرند برای اینکار، نوار را به اندازه یک خانه به سمت راست شیفت میدهیم و یک سمبول جدید را در ابتدای نوار وارد میکنیم مثلاً *. حال به خانه دوم میرویم که معادل اولین خانه برای تورینگ صورت سوال میباشد. حال ماشین تورینگ صورت سوال را قدم به قدم شبیه سازی میکنیم. حال هرگاه در نوار M' به * رسیدیم یعنی در ماشین تورینگ استاندارد(صورت سوال) سعی داشتیم که به خانه سمت چپ خانه اول نوار ببریم. پس در M' به یک ترپ استیت میرویم، به نحوی که به ازای تمام الفبای زبان، یه قانون داریم: l / a و از آنجایی که در ماشین تورینگ استاندارد(همانی که در سرکلاس گفته شده) حرکت به سمت چپ در ابتدای نوار معادل stay هست. برای همیشه در ابتدای نوار خواهیم ماند.

۲. تورینگ-ماشین‌ای داریم که از پس نوشته شدن ورودی روی نوار، در هر خانه‌ی نوار حداقل یکبار می‌توانیم بنویسیم. اثبات کنید که این ماشین با ماشین تورینگ استاندارد برابر است.

پاسخ:

برای اینکار، ابتدا، تورینگ ماشین ای که تنها میتوانیم دوبار در هر خانه بنویسیم را بررسی میکنیم و اثبات میکنیم که با تورینگ ماشین استاندارد برابر است. برای شبیه سازی تورینگ ماشین استاندارد توسعه ماشین گفته شده، هر بار کل (قسمت فعلی) نوار را در یک قسمت دست نخورد از نوار که در سمت راست قسمت فعلی قرار دارد. کپی میکنیم. برای اینکار کاراکتر به کاراکتر کپی را انجام میدهیم و علامت میزنیم که کاراکتر آیا کپی شده است یا نه. در حین وقتی به خانه ای میرسیم که هدر روی آن قرار دارد، قانون خواسته شده در آن وضعیت را اجرا میکنیم. پس بدین صورت به ازای هر قانون، قسمت فعلی نوار را همراه با اجرا اون rule (پدیت کردن خانه مورد نظر) دوباره در سمت راست کپی میکنیم. بدین صورت ماشین تورینگ را شبیه سازی میکنیم. با توجه به اینکه در این ماشین تنها اجازه داریم که در هر خانه حداقل دوبار بنویسیم (یکی برای نوشتن اولیه، بار دیگر برای علامت دار کردن هنگام کپی کردن آن). در نتیجه این کار قابل انجام هست و این ماشین با ماشین تورینگ استاندارد معادل است.

حال با دانستن اینکه ماشین تورینگ گفته شده، با تورینگ ماشین استاندارد معادل است میتوانیم نشان دهیم که write-only-TM نیز با تورینگ ماشین استاندارد معادل است:

فرایندی مشابه با ماشین تورینگ گفته شده در بالا را انجام میدهیم. فقط از آنجایی که نمیتوانیم در یک خانه دوبار بنویسیم برای اینکه بفهمیم که آن خانه کپی شده است یا نه (نقش علامت زدن در بالا را ایفا میکند). به ازای هر خانه، دو خانه در نظر میگیریم. به طوری که یکی برای نوشتن محتویات و دیگری به عنوان علامت (خلالی یا پر بودن اون خونه) در هنگام کپی.

3. نشان دهید ماشین های صفتدار معادل ماشین تورینگ استاندارد هستند.

ماشین های صفتدار مانند pda ها هستند. با این تفاوت که به جای پشته، یک صفت قرار گرفته است. صفت، در حکم یک نوار است که نمادها صرفاً از انتهای چپ نوشته و از انتهای راست خوانده می شوند (با برعکس). در واقع این ماشین یک اوتوماتی محدود با تعدادی استیت است که میتواند در هر مرحله بر اساس عنصر ورودی و استیتی که در آن قرار دارد و عنصر سر صفت استیت خود را عوض کند. این ماشین ها از یک سمبول خاص برای نشان دادن ته صفت استفاده میکنند.

پاسخ:

باید ثابت کنیم:

الف) هر ماشین تورینگ M قابل تبدیل به یک ماشین صفت Q است که همان زبان را می پذیرد.

ب) هر ماشین صفت Q قابل تبدیل به یک ماشین تورینگ M است که همان زبان را می پذیرد.

(الف)

می خواهیم ثابت کنیم برای ماشین تورینگ M ، یک ماشین صفت Q معادل داریم. صفت را در ماشین Q اینگونه در نظر می گیریم که از سمت راست به آن می توانیم پوش کنیم و از سمت چپ از آن پول کنیم.

$\text{pull} \leftarrow [a, b, c, d, e] \leftarrow \text{push}$

فرض می کنیم در ماشین M این کافیگ را داریم:

$a, b, c, \dots, h, \dots, x, y, z$

و هد خواندن بر روی h است.

در ماشین Q ، از h شروع می کنیم و در صفت پوش می کنیم. وقتی به آخر رسیدیم $\$$ را پوش می کنیم، سپس دوباره از سمت چپ شروع به پوش کردن می کنیم. پس صفت معادل برای ماشین Q به این صورت خواهد بود
(با فرض این که هد خواندن روی h قرار داشته است):

$\text{pull} \leftarrow [h, \dots, x, y, z, \$, a, b, c, \dots] \leftarrow \text{push}$

در این حالت، در ماشین صفت Q ، در سر صفت همان h قرار دارد. در واقع ماشین صفت، همان ورودی ای را در ابتدا داشت که M داشته است، با توجه به این که هد M در کجا قرار دارد، یک بار در Q کل ورودی را پیمایش می کنیم و صفت را اینگونه تشکیل می دهیم. همچنین می توانیم فرض کنیم که در ابتدا همیشه هد خواندن M در ابتدای ورودی است.

برای اثبات اینکه Q همانند M رفتار می کند، باید دو عمل M را که عبارتند از:

$a \rightarrow x, R$

$b \rightarrow y, L$

را در Q شبیه سازی کنیم. ای دو عمل، دو عملی اصلی در ماشین تورینگ هستند که عبارتند از نوشتن بر روی $tape$ و حرکت به راست و بر عکس. برای عمل اول، کافیست در Q ، ابتدا a را pull کنیم و x را پوش کنیم:

$M: [a, b, c] \longrightarrow [x, b, c]$

Q: [a, b, c, \$] ----> [b, c, \$, x]

که در بالا، بعد از این عملیات، هد خواندن در هر دو ماشین بر روی b خواهد بود. برای عملیات دوم این مراحل را در Q طی می کنیم:

۱. یک نماد # را در صفت پوش میکنیم که نشان دهنده مکان فعلی هد خواندن است.
۲. به ازای هر دو حرف پشت سر هم در صفت، یک دوتایی (x, y) در نظر می گیریم. این کار را با state های در Q می توان انجام داد. دوتایی آخر که به # منجر می شود را به طور بر عکس در صفت پوش می کنیم.
۳. دوتایی ها را از ابتدای صفت pull کرده و هر حرف را جدا جدا در صفت پوش می کنیم.
۴. نماد # را از صفت حذف می کنیم(pull).

M: [a, b, c] ----> [a, y, c]

● در ابتدا هد خواندن بر روی b بوده است.

Q:

1. [b, c, ..., \$, a] ----> [b, c, ..., \$, a, #]
2. [b, c, ..., \$, a, #] ----> [(b, c), (c, d), ..., (\$, a), #, a]
3. [(b, c), (c, d), ..., (\$, a), #, a] ----> [#, a, y, c, d, ..., \$]
4. [a, y, c, d, ..., \$]

● در این میان، b را نیز به y تغییر دادیم.

پس به ازای هر دو عمل اصلی در M ، معادل در Q می توانیم رفتار کنیم. پس هر ماشین تورینگ به ماشین صفت قابل تبدیل است که همان زبان را می پذیرد.
(ب)

فرض می کنیم ماشین صفت Q را داریم. سعی میکنیم ماشین تورینگ M معادل را بسازیم. برای این کار مثل قبل، باید دو عمل pull و push کردن در Q را در M شبیه سازی کنیم.

با پوش کردن یک حرف d در صفت، در ماشین M ، به انتهای رشته می رویم و d را اضافه می کنیم:

Q: [a, b, c] ----> [a, b, c, d]

M: [L, a, b, c, L] ----> [L, a, b, c, d, L]

برای pull کردن نیز اولین حرف را به L تبدیل می کنیم:

Q: [a, b, c] ----> [b, c]

M [□, a, b, c, □] ----> [□, □, b, c, □]

همچنین در ماشین M ، علامت □ (اولی از سمت چپ) را ابتدای ورودی در نظر می گیریم.

پس طبق الف) و ب) زبان ماشین تورینگ و صفات معادل هستند.

4. ماشین تورینگی را در نظر بگیرید که n اشاره گر برای خواندن از نوار دارد. هر حرکت در این ماشین به استیت و نمادهای زیر هر اشاره گر بستگی دارد. در هر حرکت، میتوان یک نماد در هر یک از خانه هایی که اشاره گر خوانده است نوشت، هر اشاره گر را به راست یا چپ حرکت داد یا هیچ حرکتی انجام نداد. اشاره گرها را از 1 تا n شماره گذاری می کنیم. ممکن است چند اشاره گر به یک خانه اشاره کنند؛ در این صورت نمادی که اشاره گر بزرگتر می نویسد در آن خانه قرار می گیرد. ثابت کنید این ماشین تورینگ با ماشین تورینگ استاندارد هم ارز است.(راهنمایی از ماشین تورینگ 3 نواره استفاده کنید).

پاسخ:

واضح است که این ماشین می تواند ماشین تورینگ استاندارد را شبیه سازی کند. کافی است همه اشاره گرها با هم حرکت کنند و فقط اشاره گر n ام در خانه بنویسد. این گونه انگار همان یک اشاره گر را داریم. سپس اثبات می کنیم که ماشین تورینگ می تواند ماشین n اشاره گره را شبیه سازی کند. به جای ماشین تورینگ استاندارد از ماشین تورینگ 3 نواره استفاده می کنیم و چون هم ارز هستند مشکلی پیش نمی آید. نوار اول برابر با نوار ماشین n اشاره گره خواهد بود و همان ورودی روی آن نوشه می شود. نوار دوم در ابتدای خالی است. در نوار سوم به ازای همه زیرمجموعه های {1, 2, ..., n} یک نماد جدید در نظر می گیریم و در آن خانه نمادی را می نویسیم که آن زیرمجموعه از اشاره گرها به آن اشاره می کنند. در هر مرحله در ماشین n اشاره گره به ازای هر اشاره گر، یک نماد و یک جهت حرکت خواهیم داشت. با استفاده از نوار سوم ماشین تورینگ که جایگاه هر اشاره گر را داریم در نوار دوم در جایگاه فعلی اشاره گر نماد جدید را می نویسیم. (از مکان اشاره گر اول شروع می کنیم. در این صورت اگر چند اشاره گر به یک خانه اشاره کنند، نمادی که اشاره گر بزرگتر می نویسد نهایی می شود) سپس نمادهای نوار سوم را آپدیت می کنیم تا جایگاه جدید اشاره گرها را نشان بدهد. سپس هر خانه ای از نوار دوم که خالی نیست را در نوار اول می نویسیم و در نوار دوم آن را به blank تبدیل می کنیم. بدین صورت در هر مرحله نوار اول ماشین تورینگ با ماشین گفته شده یکسان خواهد بود.

5. ماشین تورینگی با یک نوار به عنوان ورودی و یک نوار برای محاسبات داریم. در نوار اولی نمیتوان بنویسیم ولی بر روی نوار محاسبات میتوان نوشت. ثابت کنید توان محاسباتی این ماشین تورینگ با ماشین تورینگ استاندارد یکسان است.

پاسخ:

به سادگی میتوان هر حرف ورودی نوار ورودی را در نوار محاسبات کپی کرد و ادامه‌ی کار را از ابتدای نوار محاسبات پیش برد. به این طریق اثبات کردیم که هر ماشین تورینگ استانداردی با ماشین تورینگ مسئله قابل شبیه‌سازی است. حال بر عکس آن را نیز اثبات میکنیم. برای این کار کافیست از ماشین تورینگ دو نواره (که اثبات شد معادل ماشین تورینگ استاندارد است) استفاده کنیم. به همان طریق یک نوار را برای ورودی و یک نوار را برای محاسبات داریم. حال میتوان گفت که این دو ماشین تورینگ با هم از نظر توان محاسباتی، یکسان هستند.

6. ماشین تورینگی داریم که عمل چپ رفتن روی نوار را ندارد و به جای آن میتواند به اول نوار برود. ثابت کنید این ماشین تورینگ معادل ماشین تورینگ استاندارد است.

پاسخ:

ماشین تورینگ گفته شده را میتوان با ماشین تورینگ استاندارد شبیه‌سازی کرد؛ زیرا کافی است که اول نوار حرفی را اضافه کرده و تمامی ورودی را یک shift به راست دهیم. پس از آن به ازای هر اول نوار رفتن، کافی است که تا جایی که به حرف ابتدایی نرسیده‌ایم، چپ برویم. حال باید نشان دهیم که ماشین تورینگ استاندارد را میتوان به ماشین تورینگ گفته شده تبدیل کرد. از آن جایی که به جز چپ رفتن بقیه عملیاتها را داریم، پس اگر بتوانیم این عمل را به خوبی تبدیل کنیم، کار تمام است.

1. خانه فعلی را علامت میزنیم
2. به ابتدا بر میگردیم.
3. کل نوار را به راست شیفت میدهیم. درحالیکه علامت را در موقعیت فعلی اش نگه میداریم.
4. به ابتدا میرویم و به راست میرویم تا به خانه مارک شده برسیم.
پایان.

7. (امتیازی) زبان برنامه نویسی را در نظر بگیرید که نواری همانند ماشین تورینگ استاندارد دارد. الفای نوار به صورت $\{a_0, a_1, a_2, \dots, a_n\}$ است و الفای زبان شامل a_0 نمی شود. در ابتدا ورودی روی نوار نوشته شده و در ادامه آن بی نهایت a_0 می آید. دستورات زیر در این زبان تعریف شده اند:

- < : حرکت اشاره گر به سمت چپ در صورت امکان. در غیر این صورت حرکتی انجام نمی شود.
- > : حرکت اشاره گر به سمت راست
- + : اضافه کردن به نماد زیر اشاره گر:
- : کم کردن از نماد زیر اشاره گر
- a_1 به a_0 تبدیل شده، a_1 به a_2 و همینطور الى آخر. a_{n-1} به a_0 تبدیل می شود.

- : کم کردن از نماد زیر اشاره گر: a_1 به a_0 تبدیل شده، a_1 به a_2 و همینطور الى آخر. a_0 به a_{n-1} تبدیل می شود.

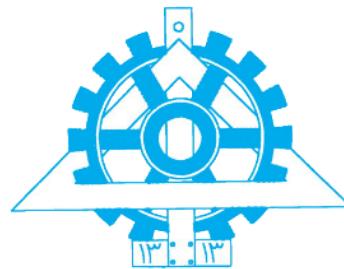
[cmds] : همانند یک حلقه while عمل می کند: تا زمانی که نماد زیر اشاره گر a_0 نیست، دستورات cmds اجرا می شوند. این حلقه ها می توانند تو در تو باشند. ثابت کنید هر برنامه به این زبان را می توان در یک ماشین تورینگ استاندارد اجرا کرد

پاسخ:

الفای ماشین تورینگ را نیز a_1 تا a_n در نظر می گیریم و به جای a_0 از blank استفاده می کنیم. از ماشین تورینگی استفاده می کنیم که قابلیت stay را نیز دارد. دستورات < و > که به سادگی در ماشین تورینگ قابل پیاده سازی هستند. کافی است همان نمادی که اشاره گر به آن اشاره می کند بازنویسی شود و اشاره گر به راست یا چپ برود. برای - نیز هر کاراکتر را با کاراکتر پایین تر از خود جایگزین کرده و اشاره گر را تکان نمی دهیم. a_0 به blank تبدیل می شود. برای + هم به همین صورت عمل می کنیم و فقط blank به a_n تبدیل می شود. برای پیاده سازی حلقه کافی است که استیت های درون حلقه را پیاده سازی کنیم و اگر در آخرین استیت اشاره گر به blank اشاره می کرد به استیت جدید برویم و درواقع دستور بیرون حلقه اجرا شود و در غیر این صورت دوباره به استیت اولی که توسط این حلقه وارد آن شدیم برگردیم.



به نام خدا



نظریه زبان‌ها و ماشین‌ها- بهار ۱۴۰۲

تمرین شماره ۹

دستیار آموزشی این مجموعه: سپهر آزردار

sepehr81sepehr@gmail.com

تاریخ تحويل: ۲۷ اردیبهشت

۱. (۱۰نمره) در ماشین تورینگ استاندارد، اگر در ابندای نوار باشیم و بخواهیم به سمت چپ حرکت کنیم، آنگاه در همان جا خواهیم ماند. ماشین تورینگ 'M' را در نظر بگیرید. این ماشین در شرایط گفته شده به ترب استیت می‌رود و در همان جا برای همیشه خواهد ماند. نشان دهد که چگونه میتوان ماشین تورینگ استاندارد را به ماشین تورینگ گفته شده تبدیل کرد به طوری که هر دو یک زبان را بپذیرند

۲. (۲۰نمره) تورینگ-ماشین‌ای داریم که از پس نوشته شدن ورودی روی نوار، در هر خانه‌ی نوار حداقل یکبار می‌توانیم بنویسیم. اثبات کنید که این ماشین با ماشین تورینگ استاندارد برابر است.

۳. (۲۰نمره) نشان دهد ماشین‌های صفتدار معادل ماشین تورینگ استاندارد هستند.
ماشین‌های صفتدار مانند pda ها هستند. با این تفاوت که به جای پشت، یک صفت قرار گرفته است. صفت، در حکم یک نوار است که نمادها صرفاً از انتهای چپ نوشته و از انتهای راست خوانده می‌شوند(با بر عکس). در واقع این ماشین یک اوتوماتای محدود با تعدادی استیت است که میتواند در هر مرحله بر اساس عنصر ورودی و استیتی که در آن قرار دارد و عنصر سر صفت استیت خود را عوض کند. این ماشین‌ها از یک سمبول خاص برای نشان دادن ته صفت استفاده می‌کنند.

۴. (۲۰نمره) ماشین تورینگی را در نظر بگیرید که n اشاره گر برای خواندن از نوار دارد. هر حرکت در این ماشین به استیت و نمادهای زیر هر اشاره گر بستگی دارد. در هر حرکت، میتوان یک نماد در هر یک از خانه‌هایی که اشاره گر خوانده است نوشت، هر اشاره گر را به راست یا چپ حرکت داد یا هیچ حرکتی انجام نداد. اشاره گر هارا از ۱ تا n شماره گذاری می‌کنیم. ممکن است چند اشاره گر به یک خانه اشاره کنند؛ در این صورت نمادی که اشاره گر بزرگتر می‌نویسد در آن خانه قرار می‌گیرد. ثابت کنید این ماشین تورینگ با ماشین تورینگ استاندارد هم ارز است. (راهنمایی از ماشین تورینگ ۳ نواره استفاده کنید.)

۵. (۱۰نمره) ماشین تورینگی با یک نوار به عنوان ورودی و یک نوار برای محاسبات داریم. در نوار اولی نمیتوان بنویسیم ولی بر روی نوار محاسبات میتوان نوشت. ثابت کنید توان محاسباتی این ماشین تورینگ با ماشین تورینگ استاندارد یکسان است.

6. (20نمره) ماشین تورینگی داریم که عمل چپ رفتن روی نوار را ندارد و به جای آن میتواند به اول نوار برود.
ثابت کنید این ماشین تورینگ معادل ماشین تورینگ استاندارد است.

7. (امتیازی 10نمره) زبان برنامه نویسی را در نظر بگیرید که نواری همانند ماشین تورینگ استاندارد دارد. الفبای نوار به صورت $\{a_0, a_1, a_2, \dots, a_{n-1}\}$ است و الفبای زبان شامل a_0 نمی شود. در ابتدا ورودی روی نوار نوشته شده و در ادامه آن بی نهایت a_0 می آید. دستورات زیر در این زبان تعریف شده اند:
< : حرکت اشاره گر به سمت چپ در صورت امکان. در غیر این صورت حرکتی انجام نمی شود.
> : حرکت اشاره گر به سمت راست
+ : اضافه کردن به نماد زیر اشاره گر:
- : کم کردن از نماد زیر اشاره گر
 a_1 به a_2 و همینطور الى اخر. a_0 به a_{n-1} تبدیل می شود.

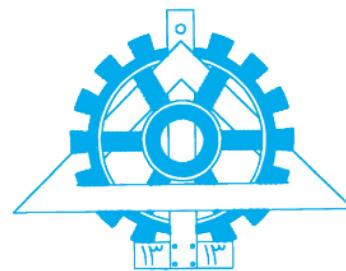
- : کم کردن از نماد زیر اشاره گر: a_1 به a_0 تبدیل شده، a_2 به a_1 و همینطور الى اخر. a_0 به a_{n-1} تبدیل می شود.

[cmds] : همانند یک حلقه while عمل می کند: تا زمانی که نماد زیر اشاره گر a_0 نیست، دستورات cmds اجرا می شوند. این حلقه ها می توانند تو در تو باشند.
ثابت کنید هر برنامه به این زبان را می توان در یک ماشین تورینگ استاندارد اجرا کرد.
(تورینگ ماشین با قابلیت stay قادری برابر با تورینگ ماشین استاندارد دارد و نیازی به اثبات آن نیست.)



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۱۰

دستیار آموزشی این مجموعه: صبا شهسواری
sabashahsvari@ut.ac.ir

تاریخ تحويل: ۷ خرداد

۱. زبان L شامل تمام DFA هایی می‌شود که زبانشان نامتناهی است. ثابت کنید L تصمیم پذیر است.

برای تشخیص DFA های عضو این مجموعه از الگوریتم زیر استفاده می‌کنیم:

فرض کنید n تعداد state های برای یک DFA خاص باشد. در صورتی که هیچ رشته‌ای با طول بیشتر از n در زبان موجود نباشد، مطمئناً زبان متناهی است. برای رشته‌های به طول بیشتر از n از روش زیر استفاده می‌کنیم:

ادعا می‌کنیم که در صورتی که رشته‌ای با طول n یا بیشتر در زبان موجود باشد، حتماً رشته‌ای با طول n تا $2n-1$ در زبان موجود است. برای اثبات از برهان خلف استفاده می‌کنیم. فرض کنید m طول کوتاه‌ترین رشته در میان رشته‌های با طول بیشتر از n است. اگر $m > 2n$ باشد می‌توانیم از pumping lemma استفاده کنیم و z را به صورت $wxy =$ بنویسیم. میدانیم wx رشته‌ای در زبان است. اما از آنجا که $n \leq |wx| \leq m$ است پس $|wy| \geq n > |wy|$ که این با فرض اولیه که m کوتاه‌ترین رشته در میان رشته‌های با طول بیشتر از n است در تناقض است.

بنابر این کافی است برای بررسی متناهی بودن زبان یک DFA عضویت تمام رشته‌های با طول بین n تا $2n-1$ را در آن زبان بررسی کنیم. در صورتی که چنین رشته‌ای وجود نداشته باشد، زبان متناهی است و در غیر این صورت زبان نامتناهی است. چون در تعداد گام‌های محدودی می‌توان این بررسی را انجام داد پس زبان L تصمیم پذیر است.

۲. زبان L شامل زوج‌های $\langle G, B \rangle$ است که در آن G یک گرامر مستقل از متن و B یک متغیر در آن است به طوری که B در اشتقاق حداقل یکی از رشته‌هایی که توسط G تولید می‌شود استفاده شده است. ثابت کنید این زبان تصمیم پذیر است.

گرامر G را به شکل زیر می‌سازیم:

تمام قواعد G که در سمت چپشان B وجود دارد را حذف می‌کنیم و در قواعدی که سمت راستشان B وجود دارد، به جای B از یک نماد جدید مانند B' استفاده می‌کنیم.

حال اشتراک زبان G' را با زبان $(\Sigma^* B') \cup (B' \Sigma^*)$ حساب می‌کنیم. می‌دانیم اشتراک یک زبان منظم و یک گرامر مستقل از متن یک زبان مستقل از متن است و با یک الگوریتم پایان پذیر می‌توان

اشتراک آنها را حساب کرد. اگر اشتراک این دو زبان تهی باشد به این معناست که B در اشتقاق هیچ رشته استفاده نشده پس زبان رد می‌شود و در غیر این صورت پذیرفته می‌شود.

3. عبارت منظم R زبان‌هایی را توصیف می‌کند که حداقل شامل یک رشته مانند w هستند به طوری که 111^* زیررشته‌ای از w است. ثابت کنید تشخیص این که زبان $'R$ چنین ویژگی دارد یا نه تصمیم پذیر است.

زبان عبارت منظم $'R$ را می‌توان به شکل $('R \cap L(R))^*$ نوشت. برای تصمیم گیری این زبان از الگوریتم زیر استفاده می‌کنیم:

1- یک DFA به نام A می‌سازیم که 111^* را می‌پذیرد.

2- از اشتراک A و R یک DFA به نام B می‌سازیم. $L(B) = L(A) \cap L(R)$. (می‌دانیم می‌توان با الگوریتم پایان پذیر DFA حاصل از اشتراک دو DFA را تشکیل داد).

3- ماشین تورینگ M که می‌تواند A را **decide** کند ساخته و آن را روی B اجرا می‌کنیم. اگر M بپذیرد به این معناست که $L(A) \cap L(R) = \emptyset$ بوده پس $'R$ این ویژگی را نداشته و رد می‌شود. اگر M رد کند یعنی $'R$ این ویژگی را داشته و پذیرفته می‌شود.

4. فرض کنید A و B دو زبان **turing-recognizable** باشند. تشخیص پذیر یا تشخیص ناپذیر بودن زبان $A-B$ را ثابت کنید.

با استفاده از یک مثال نقض ثابت می‌کنیم $B - A$ تشخیص پذیر نیست. فرض کنید $A = A_{TM}^{\Sigma^*}$ و $B = A_{TM}^{\Sigma^*}$ باشد. می‌دانیم A_{TM} تصمیم پذیر نیست. $B - A$ مکمل زبان A_{TM} می‌شود. می‌دانیم اگر یک زبان و مکملش هر دو تشخیص پذیر باشند، آن زبان حتما تصمیم پذیر است در نتیجه $B - A$ تشخیص ناپذیر است.

5. ثابت کنید این مسئله که آیا ماشین تورینگ M فقط رشته‌هایی که **Palindrome** هستند را می‌پذیرد یا خیر، تصمیم ناپذیر است.

با استفاده از یک مثال نقض ثابت می‌کنیم این زبان تصمیم پذیر نیست.

فرض کنید تصمیم پذیر باشد پس یک ماشین تورینگ مانند D وجود دارد که آن را می‌پذیرد. با استفاده از این ماشین یک ماشین تورینگ N را به شکل زیر طراحی می‌سازیم:

ماشین N ورودی $\langle M, w \rangle$ که M توصیف یک ماشین تورینگ و w رشته ورودی است را دریافت می‌کند و ماشین تورینگ R را می‌سازد. ماشین R ورودی $\langle M, w \rangle$ را دریافت می‌کند و اگر w یک رشته palindrome نباشد آن را رد می‌کند و در غیر این صورت، ماشین M را روی w اجرا می‌کند اگر M پذیرفت، ماشین R نیز می‌پذیرد و در غیر این صورت رد می‌کند. پس از ساخت ماشین R ، توصیف آن را به عنوان ورودی به ماشین D می‌دهد. اگر D پذیرفت، N نیز می‌پذیرد و اگر D رد کرد ماشین N نیز رد می‌کند.

چون فرض کرده بودیم D یک decider است پس حتماً یا می‌پذیرد یا رد می‌کند و به همین دلیل ماشین N نیز حتماً یا می‌پذیرد یا رد می‌کند (در نهایت متوقف می‌شود). پس ماشین N یک decider برای زبان A_{TM} است. در حالی که در کلاس اثبات شده زبان A_{TM} تصمیم پذیر نیست پس فرضی که داشتیم خلط است. یعنی D نمی‌تواند وجود داشته باشد و این زبان تصمیم ناپذیر است.

6. تصمیم پذیری زبان‌های زیر را بررسی کنید.

(الف) زبان L شامل توصیف ماشین تورینگ M و رشته w است به طوری که ماشین M در پردازش رشته w بیشتر از یک بار وارد یک state می‌شود.

این زبان تصمیم پذیر است. می‌دانیم تعداد state های یک ماشین تورینگ محدود است. اگر تعداد state های ماشین تورینگی که در ورودی داده شده N باشد، این ماشین حداقل می‌تواند $2N-1$ گام بردارد بدون آن که دو بار وارد یک state شود و در گام $2N$ (طبق اصل لانه کبوتری) حتماً وارد یک state تکراری می‌شود. می‌توانیم ماشین تورینگی به شکل زیر برای decide کردن این زبان طراحی کنیم:

می‌دانیم ماشین تورینگ دونواره قدرت مشابهی با ماشین تورینگ تکنواره دارد پس از یک ماشین دو نواره به نام N استفاده می‌کنیم، به ازای ورودی $\langle M, w \rangle$ ، ابتدا روی نوار دوم شماره تمام state های ماشین M را دو بار می‌نویسیم سپس M را روی w اجرا می‌کنیم. هر بار که M به state شماره i رفت در نوار دوم دنبال عدد i می‌گردیم اگر پیدا شد آن را پاک کرده و ادامه می‌دهیم و اگر پیدا نشد یعنی M می‌خواهد برای بار دوم وارد آن state شود و پذیرفته می‌شود. اگر ماشین M رشته w را پذیرفت یا رد کرد آن گاه N ورودی $\langle M, w \rangle$ رد می‌کند.

(ب) زبان شامل تمام توصیف های ماشین‌های تورینگ مانند $\langle M \rangle$ که M روی h ورودی به طول k متوقف می‌شود.

این زبان تصمیم پذیر است. می‌توانیم ماشین تورینگ پذیرنده این زبان را به شکل زیر تعریف کنیم:

ماشین تورینگ T به ازای ورودی $\langle M \rangle$ ، ماشین M را روی تمام رشته‌ها به طول k اجرا می‌کند. اگر M هر یک از این رشته‌ها را رد کند یا هرگز متوقف نشود، ماشین T ورودی را رد می‌کند و اگر M تمام این رشته‌ها را بپذیرد، ماشین T ورودی را می‌پذیرد. چون تعداد رشته‌ها به طول k محدود است پس در نهایت ماشین T متوقف می‌شود و M را decide می‌کند.

7. ثابت کنید زبان‌های تشخیص پذیر تحت عمل concatenation بسته هستند.

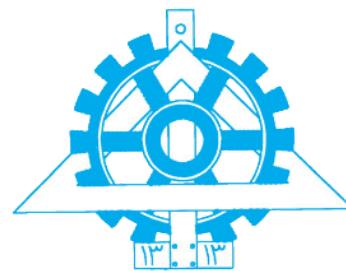
دو زبان تشخیص پذیر A و B را در نظر بگیرید. چون A تشخیص پذیر هستند پس دو ماشین تورینگ مانند وجود دارند که می‌توانند به ترتیب A و B را تشخیص دهند. ماشین تورینگ M_{AB} را برای تشخیص زبان AB به شکل زیر می‌سازیم:

- ۱- ورودی را دریافت می‌کنیم و آن را به دو زیر ورودی تقسیم می‌کنیم
- ۲- با استفاده از ماشین M_A ، ورودی اول را بررسی می‌کنیم. اگر قبول شود، به مرحله بعدی می‌رویم و اگر قبول نشود، نیز آن را رد می‌کند.
- ۳- با استفاده از ماشین M_B ، ورودی دوم را بررسی می‌کنیم. اگر قبول شود، به مرحله بعدی می‌رویم و اگر قبول نشود، نیز آن را رد می‌کند.
- ۴- اگر هر دو ورودی قبول شوند، ورودی را قبول می‌کنیم و در غیر این صورت رد می‌شود.



به نام خدا

نظریه زبان‌ها و ماشین‌ها - بهار ۱۴۰۱



تمرین شماره ۱۰

دستیار آموزشی این مجموعه: صبا شهسواری

sabashahsavari@ut.ac.ir

تاریخ تحويل: ۷ خرداد

۱. زبان L شامل تمام DFA هایی می‌شود که زبانشان نامتناهی است. ثابت کنید L تصمیم پذیر است. (20 نمره)

۲. زبان L شامل زوج‌های $\langle G, B \rangle$ است که در آن G یک گرامر مستقل از متن و B یک متغیر در آن است به طوری که B در اشتقاق حداقل یکی از رشته‌هایی که توسط G تولید می‌شود استفاده شده است. ثابت کنید این زبان تصمیم پذیر است. (10 نمره)

۳. عبارت منظم R زبان‌هایی را توصیف می‌کند که حداقل شامل یک رشته مانند w هستند به طوری که ۱۱۱ زیررشته‌ای از w است. ثابت کنید زبان R تصمیم پذیر است. (15 نمره)

۴. فرض کنید A و B دو زبان turing-recognizable باشند. تشخیص پذیر یا تشخیص ناپذیر بودن زبان $A-B$ را ثابت کنید. (15 نمره)

۵. ثابت کنید این مسئله که آیا ماشین تورینگ M فقط رشته‌هایی که Palindrome هستند را می‌پذیرد یا خیر، تصمیم ناپذیر است. (20 نمره)

۶. تصمیم پذیری زبان‌های زیر را بررسی کنید. (20 نمره)

الف) زبان L شامل توصیف ماشین تورینگ M و رشته w است به طوری که ماشین M در پردازش رشته w بیشتر از یک بار وارد یک state می‌شود.

ب) زبان شامل تمام توصیف‌های ماشین‌های تورینگ مانند $\langle M \rangle$ که M روی h ورودی به طول k متوقف می‌شود.

۷. (امتیازی) ثابت کنید زبان‌های تشخیص پذیر تحت عمل concatenation بسته هستند. (10 نمره)

به نام خدا



نظریه زبان‌ها و ماشین‌ها – بهار ۱۴۰۲

تمرین شماره ۱۱

دستیار آموزشی این مجموعه: سامان اسلامی نظری

Saman.Eslami78@gmail.com

تاریخ تحويل: ۱۴ خرداد (صفحه درس)

۱. فرض می‌کنیم که زبان EQ_{CFG} تصمیم‌پذیر است؛ زبان ALL_{CFG} را به آن کاهش می‌دهیم:

ماشین تورینگ M را برای زبان ALL_{CFG} به صورت زیر می‌سازیم:

$M = "به ازای ورودی \langle G \rangle :$

۱. یک گرامر H که زبان آن Σ^* را می‌سازیم.

۲. ماشین تصمیم‌گیرنده EQ_{CFG} را به ازای ورودی $\langle G, H \rangle$ اجرا می‌کنیم.

۳. در صورتی که آن را قبول کرد ما نیز ورودی را قبول می‌کنیم. در غیر این صورت ورودی را رد می‌کنیم."

۲. فرض می‌کنیم مسئله مذکور توسط ماشین تورینگ H تصمیم‌پذیر است. مسئله A_{TM} را به این مسئله، به روش زیر، کاهش می‌دهیم:

$S = "به ازای ورودی \langle M, w \rangle :$

۱. از M برای ساختن ماشین تورینگ دو نواره به صورت زیر استفاده می‌کنیم:

$T = "به ازای ورودی \langle x \rangle :$

۱. ماشین M را روی نوار اول شبیه‌سازی می‌کنیم.

۲. اگر شبیه‌سازی موفق بود، روی نوار دوم یک کاراکتر می‌نویسیم."

۳. ماشین H را روی ورودی $\langle T, w \rangle$ اجرا می‌کنیم. اگر آن را قبول کرد، ورودی را می‌پذیریم و در غیر این صورت آن را رد می‌کنیم."

۴. بدون از دست دادن کلیت مسئله، فرض می‌کنیم که الفبای این زبان تنها شامل ۱ می‌شود. بنابراین مسئله PCP به صورت زیر خواهد بود:

$$P = \left\{ \left[\frac{1^{a_1}}{1^{b_1}} \right], \left[\frac{1^{a_2}}{1^{b_2}} \right], \dots, \left[\frac{1^{a_n}}{1^{b_n}} \right] \right\}$$

ماشین تورینگ زیر را برای تصمیم‌گیری این زبان می‌سازیم:

$$M = "برای ورودی \langle P \rangle"$$

۱. اگر به ازای یک i -ای، $a_i = b_i$ وجود داشت، ورودی را قبول کن.

۲. اگر به ازای یک i که $b_j < a_i$ یک j وجود داشت که $b_j < a_i$ بود آنگاه ورودی را قبول کن. در غیر این صورت ورودی را رد کن."

دلیل بخش اول واضح است. برای بخش دوم، برای حل مسئله، تنها کافیست با استفاده از کسر $\frac{a}{b}$ تعداد یک‌های صورت کسرها را به اندازه $a - b$ بیشتر از تعداد یک‌های مخرج کسر کرده و سپس این اختلاف را با استفاده از کسر $\frac{a}{b} - \frac{b}{a}$ جبران می‌کنیم. از آنجا که در هر حالت این ماشین تورینگ توقف می‌کند، زبان داده شده تصمیم‌پذیر است.

۵. اثبات می‌کنیم که $T \leq_m A_{tm}$ برای اینکار نگاشت $\langle M, w \rangle$ به $\langle M' \rangle$ را ارائه می‌دهیم:

ماشین M' : "به ازای ورودی x

۱. در صورتی که $x \neq ab$ یا $x \neq ba$ باشد، به استیت reject می‌رویم.

۲. در صورتی که ورودی برابر ab باشد، آن را قبول می‌کنیم.

۳. در صورتی که ورودی برابر ba باشد، ابتدا ماشین تورینگ M را به ازای ورودی w اجرا کرده، و اگر آن را قبول نماید به استیت accepting می‌رویم. در غیر این صورت به استیت reject می‌رویم."

واضح است که اگر $\langle M, w \rangle \in A_{TM}$ آنگاه $\langle M, w \rangle$ را قبول کرده و $L(M') = \{ab, ba\}$ و در این صورت T نیز $L(M')$ را قبول می‌کند. اما اگر $\langle M, w \rangle \notin A_{TM}$ آنگاه $\{ab, ba\} \notin L(M')$ و T نیز $L(M') = \{ab, ba\}$ را قبول نماید. بنابراین، این نگاشت صحیح می‌باشد

۵. مسئله PCP را به این زبان کاهش می‌دهیم. یک نمونه از این مسئله به صورت $\{ \left[\frac{t_1}{b_1} \right], \left[\frac{t_2}{b_2} \right], \dots, \left[\frac{t_k}{b_k} \right] \}$ در نظر می‌گیریم. گرامر زیر را متناظر با این نمونه از PCP می‌سازیم.

$$\begin{aligned} S &\rightarrow S_1 | S_2 \\ S_1 &\rightarrow t_1 S_1 a_1 | \dots | t_k S_1 a_1 | t_1 a_1 | \dots | t_k a_k \\ S_2 &\rightarrow b_1 S_2 a_1 | \dots | b_k S_2 a_1 | b_1 a_1 | \dots | b_k a_k \end{aligned}$$

در اینجا کاراکترهای a_k, a_2, a_1, \dots کاراکترهای اضافی هستند که در مسئله PCP-مان موجود نیستند. حال اگر این گرامر مبهم باشد، یک رشته W را به دو صورت از قوانین بالا می‌توان بدست آورد. بدون از دست دادن کلیت مسئله، فرض می‌کنیم که با قانون $S_1 \rightarrow S$ شروع کرده و به دو صورت رشته مورد نظر را می‌سازیم. در صورتی که رشته موردنظر توسط گرامر پذیرفته شود، انتهای آن شامل کاراکترهای جدید خواهد بود؛ از آنجا که تنها به یک صورت از طریق $S_1 \rightarrow S$ می‌توان این کاراکترهای جدید را ساخت، بنابراین تنها یک راه از طریق قانون $S_1 \rightarrow S$ برای ساخت رشته W خواهیم داشت. پس اگر گرامر مبهم باشد، یک درخت اشتاقاً از قانون $S_1 \rightarrow S$ و درخت دیگر از قانون $S_2 \rightarrow S$ استفاده خواهد کرد. حال در صورتی که این گرامر مبهم باشد یک رشته مانند W وجود خواهد داشت که یک بار از راه قانون اول و بار دیگر از راه قانون دوم پذیرفته می‌شود. در این رشته، زیررشته سمت چپ آن که شامل کاراکترهای PCP است نیز با یکدیگر یکسان خواهد بود و این زیررشته نشان‌دهنده یک راه حل برای مسئله PCP است.

در نهایت با ساخت گرامری که گفته شد، می‌توان جواب مسئله PCP داده شده را بدست آورد. اگر $AMBIG_{CFG}$ بگوید که گرامر ورودی مبهم است، مسئله PCP قابل حل و در صورتی که بگوید مبهم نیست، مسئله PCP غیرقابل حل خواهد بود. بنابراین مسئله PCP که یک مسئله *undecidable* است به این زبان کاهش داده شد. پس این زبان نیز *undecidable* است.

۶. فرض می‌کنیم مسئله A_{TM} توسط ماشین تورینگ H_1 قابل شناسایی باشد. ماشین تورینگ H_2 را می‌سازیم که به ازای ورودی $\langle n \rangle$ مسئله $1 + 3x$ را اجرا می‌کند؛ واضح است که اگر به ازای n دنباله در جایی به ۱ برسد متوقف شده و ورودی را قبول می‌کند؛ در غیر این صورت تا ابد در چرخه گیر خواهد کرد. ماشین تورینگ H_3 را می‌سازیم که به ازای ورودی $\langle H_2, n \rangle$ ماشین H_1 را اجرا کرده و اگر آن را قبول کرد، ورودی را قبول می‌کند و در غیر این صورت آن را رد می‌کند (در اصل با استفاده از ماشین تصمیم‌گیرنده A_{TM} مسئله $1 + 3x$ را به ازای یک ورودی خاص حل کردیم). اکنون ماشین H_4 را تعریف می‌کنیم که در آن به ازای هر ورودی، ماشین H_3 را با n از ۱ تا بینهایت اجرا می‌کند؛ در صورتی که یک عددی وجود داشته باشد که به ازای آن، دنباله به یک ختم نشود، ماشین از کار ایستاده و به یک *accepting state* می‌رود. در غیر این صورت ماشین هیچگاه از کار نخواهد ایستاد. در نهایت ماشین H_5 را تعریف می‌کنیم که به ازای هر ورودی، ماشین H_4 را با همان ورودی خودش (دقت کنید که برای ماشین H_4 مقدار ورودی اهمیتی ندارد) به H_1 می‌دهیم؛ اگر H_1 آن را پذیرفت، می‌فهمیم که عددی وجود دارد که به ازای آن دنباله $1 + 3x$ به ۱ ختم نمی‌شود؛ اگر H_1 آن را نپذیرفت، یعنی دنباله $1 + 3x$ به ازای تمام اعداد طبیعی به ۱ ختم می‌شود. ماشین H_5 جواب مسئله است.

۷. زبان E_{TM} نشان‌گر ماشین‌ها TM -ای است که زبانشان تهی است. می‌دانیم این زبان تصمیم‌پذیر نیست. مسئله مذکور در صورت سوال را به صورت زبان زیر نشان می‌دهیم:

$$USELESS_{TM} = \{\langle M, q \rangle \mid q \text{ is a useless state in } M\}$$

فرض می‌کنیم که $USELESS_{TM}$ تصمیم‌پذیر بوده و توسط ماشین H حل می‌شود. زبان E_{TM} را به این زبان کاهش می‌دهیم: $\langle M, q_{accepting} \rangle$. ماشین H را با ورودی M به صورت $q_{accepting}$ (accepting) ماشین ورودی M را با ورودی $\langle M, q_{accepting} \rangle$ ازای هر استیت پذیرنده (accepting) می‌شود. آنگاه متوجه می‌شویم که اجرا می‌کنیم. در صورتی که H مشخص کند تمام M بی‌فایده هستند، آنگاه متوجه می‌شویم که $L(M) = \emptyset$. بنابراین بدین صورت توانستیم مسئله E_{TM} تصمیم‌پذیر نیست؛ پس فرض خلف باطل بوده و $USELESS_{TM}$ نیز تصمیم‌پذیر نمی‌باشد.

۸. نشان می‌دهیم که اگر این تابع قابل محاسبه باشد، آنگاه A_{TM} نیز decidable خواهد بود. فرض می‌کنیم تابع BB توسط ماشین F محاسبه می‌شود. ماشین S را برای A_{TM} به صورت زیر می‌سازیم:

$$\text{ماشین } S = " \text{به ازای ورودی } \langle M, w \rangle : \text{ماشین } M \text{ را روی الفبای نوار } \Gamma \text{ به صورت زیر می‌سازیم:}$$

۱. ماشین M_w را روی الفبای نوار Γ به صورت زیر می‌سازیم:

$$" \text{به ازای هر ورودی: } M_w$$

۲. ماشین M را روی ورودی w شبیه‌سازی کن. در این حین تعداد استیت‌هایی که در حین شبیه‌سازی رد می‌شوند را ذخیره کن.

۳. اگر M محاسباتش تمام شد (halt کرد)، به تعداد استیت‌هایی که در مرحله قبل برای شبیه‌سازی شمردیم، عدد یک روی نوار بنویس.

۴. با استفاده از ماشین F مقدار $k = BB(k)$ را بدست آورد. k در اینجا تعداد استیت‌های M_w است.

۵. M را به ازای w به اندازه b قدم اجرا کن.

۶. در صورتی که M با این تعداد قدم اجرا ورودی را قبول کرد، ما نیز به accepting استیت می‌رویم. در غیر اینصورت به یک rejecting استیت می‌رویم.

در صورتی که M ورودی w را قبول کند، آنگاه M_w به تعداد قدم‌های اجرای M روی نوار، یک می‌نویسد. علاوه بر آن، طبق تعریف BB می‌دانیم که مقدار b از تعداد یک‌هایی که روی نوار نوشته شده بیشتر یا برابر آن است (چون از بین تمام ماشین تورینگ‌هایی که اندازه M_w استیت دارند، b نشان‌دهنده بیشترین تعداد یک‌های روی نوار این ماشین‌هاست؛ تعداد یک‌های

روی نوار در انتهای محسسات نیز قطعاً از تعداد مراحل اجرای یک ماشین تورینگ بیشتر نخواهد بود). بنابراین S ماشین M را به اندازه کافی اجرا می‌کند تا متوجه شود که ورودی را قبول می‌کند و خودش نیز ورودی را قبول کند. اگر M ورودی را قبول نکند، S نیز هیچگاه قبول کردن M را نمیده و ورودی را رد می‌کند.

به نام خدا



نظریه زبان‌ها و ماشین‌ها – بهار ۱۴۰۲
تمرین شماره ۱۱
دستیار آموزشی این مجموعه: سامان اسلامی نظری
Saman.Eslami78@gmail.com
تاریخ تحويل: ۱۴ خرداد (صفحه درس)



۱. ثابت کنید زبان EQ_{CFG} تصمیم‌پذیر^۱ نیست. این زبان نشان می‌دهد آیا دو گرامر داده شده زبان یکسانی تولید می‌کنند یا خیر. (از تصمیم‌نایپذیری زبان ALL_{CFG} استفاده کنید)
۲. یک ماشین تورینگی با دو نوار را فرض کنید. اثبات کنید مسئله اینکه آیا این ماشین تورینگ در حین اجرا روی نوار دومش یک کاراکتر non-blank را می‌نویسد یا خیر، یک مسئله Undecidable است.
۳. ثابت کنید مسئله PCP با مجموعه Σ که تنها یک عضو دارد، decidable است.
۴. ثابت کنید زبان زیر undecidable است:

زبان T شامل تمام ماشین تورینگ‌های M -ای است که M در آن تنها در صورتی رشته w را قبول می‌کند که w^R را نیز قبول کند.

۵. ثابت کنید که زبان زیر Turing-undecidable است.

$$AMBIG_{CFG} = \{\langle G \rangle \mid G \text{ is an ambiguous CFG.}\}$$

راهنمایی: مسئله PCP را به این زبان کاهش دهید. یک نمونه به صورت $P = \left\{ \left[\frac{t_1}{b_1} \right], \left[\frac{t_2}{b_2} \right], \dots, \left[\frac{t_k}{b_k} \right] \right\}$ را به گرامر

$$\begin{aligned} S &\rightarrow S_1 | S_2 \\ S_1 &\rightarrow t_1 S_1 a_1 | \dots | t_k S_1 a_1 | t_1 a_1 | \dots | t_k a_k \\ S_2 &\rightarrow b_1 S_2 a_1 | \dots | b_k S_2 a_1 | b_1 a_1 | \dots | b_k a_k \end{aligned}$$

به طوری که a_1, a_2, \dots, a_k ترمinal‌های جدید نسبت به حروف مسئله PCP هستند.

۶. تابع زیر را در نظر بگیرید:

$$f(x) = \begin{cases} 3x + 1 & \text{for odd } x \\ x/2 & \text{for even } x \end{cases}$$

دامنه این تابع، اعداد طبیعی می‌باشد. حالتی را در نظر بگیرید که در آن با یک x شروع کرده و یک دنباله به صورت $x, f(x), f(f(x)), \dots$ بررسیم ادامه می‌دهیم. محاسبات کامپیوتری نشان داده‌اند

که برای اعداد طبیعی از یک تا عددی بسیار بزرگ، این دنباله از اعداد در نهایت به یک ختم می‌شود. با این حال این پرسش که آیا به ازای تمام اعداد طبیعی، این دنباله به یک ختم می‌شود یا نه بی‌پاسخ مانده است؛ این مسئله، به مسئله $3x + 1$ معروف است. فرض کنید که مسئله A_{TM} یک مسئله تصمیم‌پذیر باشد؛ با این فرض اثبات کنید که مسئله $3x + 1$ نیز تصمیم‌پذیر است.

۷. یک استیتیت بی‌فایده در یک ماشین تورینگ، استیتیتی است که به ازای هیچ رشته ورودی، ماشین وارد این استیت نمی‌شود. اثبات کنید که مسئله مشخص کردن وجود استیت بی‌فایده در یک ماشین تورینگ، یک مسئله undecidable است.

۸. (امتیازی) برای تمام ماشین تورینگ‌های این مسئله، زبان نوار را به صورت $\{0,1\}^*$ فرض کنید. تابع BB (busy beaver) به صورت $\mathcal{N} \rightarrow \mathcal{N}$ به طوریکه در ادامه توضیح داده شده، تعریف می‌شود؛ به ازای هر مقدار از k . تمام ماشین تورینگ‌های k -استیتی را در نظر بگیرید که با شروع از یک نوار خالی، در نهایت halt می‌کنند. $BB(k)$ را حداکثر تعداد یک‌ها میان تمام ماشین تورینگ‌ها در نظر می‌گیریم که روی نوار باقی می‌ماند. اثبات کنید این تابع قابل محاسبه نمی‌باشد.



تاریخ تحويل: ۱۴۰۲/۳/۲۴

- ۱) درستی یا نادرستی هر یک از عبارات زیر را با ذکر دلیل مشخص کنید. (۱۲ نمره)
- الف) اگر مساله B از کلاس مساله NP -Complete باشد و بدانیم $B \leq A$ آنگاه A در کلاس NP -Complete خواهد بود.
- ب) اگر مساله A از کلاس NP -Complete باشد و بدانیم $A \leq B$ آنگاه B در کلاس NP -Complete خواهد بود.
- ج) هر مساله NP -Complete را می‌توان در زمان چندجمله‌ای به هر مساله NP -Complete دیگر کاهش داد.
- د) اگر داشته باشیم $P = NP$ آنگاه به ازای هر زبان A که در دسته P قرار دارد ($A \neq \emptyset, \sum^* \in P$) در دسته NP -Complete نیز قرار خواهد داشت.

پاسخ:

- الف) نادرست؛ می‌توان به جای A یک مساله کلاس NP را قرار دهیم پس لزوماً A در کلاس NP -Complete نخواهد بود.
- ب) نادرست؛ A در کلاس NP نیز قرار دارد و هر مساله NP را می‌توان در زمان چندجمله‌ای به یک مساله NP -Hard کاهش داد ولی لزوماً B در کلاس NP قرار ندارد پس نمی‌توان گفت در کلاس NP -Complete قرار دارد.
- ج) درست؛ هر مساله NP -Complete، در کلاس NP نیز قرار دارد پس می‌توان آن را به هر مساله NP -Complete دیگر کاهش داد.

- د) درست؛ اگر $P = NP$ باشد پس A در دسته NP نیز قرار می‌گیرد و تمامی مسائل NP را می‌توان در زمان چندجمله‌ای به یکدیگر کاهش داد پس A در دسته مسائل NP -Complete نیز قرار می‌گیرد.
- (2) فرض کنید a, b, c, p اعداد مثبت باینری هستند؛ ثابت کنید که $ModeP$ در دسته مسائل P قرار دارد. (۱۲ نمره)
(راهنمایی: $((a^2)^2)_2 = ((a^2)^2)_2$)

$$ModeP = \{<a, b, c, p> \mid a^b = c \pmod{p}\}$$

پاسخ:

الگوریتم زیر را در نظر بگیرید که ModeP را تصمیم گیری می‌کند:

$$b = b_1 b_2 b_3 \dots b_n$$

1. let $T = 1$ and $n = \lceil \log_2 b \rceil$ (if $b = 2^k$, $n = \lceil \log_2 b \rceil + 1$)

2. for $i = 1$ to n :

if $b_i = 1$, $T = (a(T^2)) \pmod{p}$

if $b_i = 0$, $T = (T^2) \pmod{p}$

3. return $T \pmod{p}$

4. if $T = c \pmod{p}$ accept, otherwise reject.

طبق راهنمایی داده شده، به ازای هر 1 دیده شده در نمایش دودویی عدد، T را به توان دو رسانده و در a ضرب می‌کنیم و به ازای هر 0 دیده شده در نمایش دودویی عدد، T را به توان دو می‌رسانیم (دلیل این امر واضح است). در نهایت با مقایسه T نهایی با c می‌توانیم تصمیم بگیریم که accept یا reject. حال به تحلیل مرتبه زمانی این الگوریتم می‌پردازیم؛ فرض کنید اعداد a, b, c, p حداقل m بیتی باشند ($n \leq m$)، مرتبه زمانی ضرب یا تقسیم (همچنین تعیین باقیمانده) دو عدد m بیتی $O(m^2)$ است و اینکار در یک حلقه به تعداد n باز انجام می‌شود پس پیچیدگی زمانی کل برابر P با $(O(m^3) \times n) = O(m^3)$ است. پیچیدگی این الگوریتم نسبت به ورودی چندجمله‌ای است پس این مساله در دسته مسائل P قرار می‌گیرد.

(3) یک دور "گذر دوبل از رئوس" در گراف بدون جهت G ، دوری است که از تمامی راس‌های G دقیقاً دوبار می‌گذرد. ثابت کنید مساله تشخیص وجود دور "گذر دوبل از رئوس" در گراف G ، یک مساله NP-Hard است. (14 نمره)

پاسخ:

برای اثبات NP-Hard بودن این مساله، مساله‌ی دور همیلتونی که یک مساله‌ی NP-Complete است را به آن کاهش می‌دهیم. فرض کنید گراف G ورودی مساله دور همیلتونی باشد، می‌خواهیم آن را به گراف H که ورودی مساله تشخیص وجود دور "گذر دوبل از رئوس" است تبدیل کنیم؛ برای اینکار ابتدا گراف H را برابر گراف G قرار می‌دهیم و سپس به ازای هر راس v در گراف H دو راس v' و v'' به این گراف اضافه می‌کنیم و سه یال " v, v', v'' " را بین سه راس v و v' و v'' قرار می‌دهیم. حال ثابت می‌کنیم گراف G دور همیلتونی دارد اگر و تنها اگر گراف H دور "گذر دوبل از رئوس" داشته باشد. اگر دور $v \rightarrow v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_n \rightarrow v_1$ را یک دور همیلتونی در گراف G در نظر بگیریم، دور "گذر دوبل از رئوس" متناظر آن در گراف H دور $v \rightarrow v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_n \rightarrow v_1$ است. برای اثبات طرف دیگر، اگر دور "گذر دوبل از رئوس" در گراف H پیدا شود با حذف رئوس v' و v'' اضافه شده، یک دور همیلتونی در گراف G بدست می‌آید.

(4) مساله‌ی مقابل را در نظر بگیرید: در شهر پهلوانان، هر پهلوان عضو حداقل یک باشگاه است. شهردار شهر برای افزایش تمرکز پهلوانان فصد دارد تعدادی باشگاه را تعطیل کند به طوری که پس از تعطیلی، هر پهلوان هنوز عضو حداقل یک باشگاه تعطیل نشده باشد. ورودی مساله، لیست پهلوانان، لیست باشگاه‌ها، لیست اعضای هر باشگاه و عدد k می‌باشد. آیا شهردار می‌تواند k باشگاه را طوری انتخاب کند که پس از تعطیلی آنها هنوز هر پهلوان عضو حداقل یک باشگاه باشد؟ ثابت کنید این مساله NP-Complete است. (راهنمایی: می‌توانید از NP-Complete مساله [set cover](#) استفاده کنید). (17 نمره)

پاسخ:

ابتدا باید ثابت کنیم مجموعه NP است. برای اینکار، گواهی را لیست k باشگاهی در نظر می‌گیریم که قرار است تعطیل شوند و verifier به ازای هر پهلوان بررسی می‌کند که آیا هنوز عضو باشگاهی است که در لیست k باشگاه تعطیل شده نباشد؛ اگر این شرط برای همه‌ی پهلوانان صدق کرد، accept می‌کند و در غیر اینصورت reject می‌کند. بدیهی است اینکار در زمان چندجمله‌ای قابل انجام است پس مساله در دسته NP قرار دارد.

در مساله پوشش مجموعه، یک مجموعه متناهی U و خانواده F از زیرمجموعه‌های U داده شده است به طوری که اجتماع این زیرمجموعه‌ها برابر U است. آیا می‌توان با انتخاب k یا کمتر زیرمجموعه از F ، U را پوشش داد؟ برای اثبات NP-Hard بودن مساله پهلوانان، مساله پوشش مجموعه را به آن کاهش می‌دهیم و ورودی پوشش مجموعه را به ورودی مساله پهلوانان تبدیل می‌کنیم. اعضای U معادل پهلوانان شهر هستند و به ازای هر خانواده F یک باشگاه می‌سازیم و اعضای باشگاه، اعضای زیرمجموعه مربوطه خواهد بود. در نهایت مقدار $k_{\text{set cover}}$ را برابر با $|F| - k_{\text{pahlevanan}}$ قرار می‌دهیم؛ بدیهی است این کاهش در زمان چندجمله‌ای قابل انجام است. حال نشان می‌دهیم پاسخ مساله پهلوانان به است اگر و تنها اگر مساله vertex cover پاسخ بله داشته باشد.

طرف اول: اگر ورودی دارای پوشش رئوس به اندازه $k_{\text{set cover}}$ باشد، آنگاه مساله پهلوانان پاسخ بله دارد؛ در این حالت اگر همه‌ی باشگاه‌ها به جز $k_{\text{set cover}}$ باشگاه تعطیل شوند هنوز همه‌ی پهلوانان عضو حداقل یک باشگاه (خانواده) هستند. طرف دوم: اگر بتوان با تعطیلی $k_{\text{set cover}}$ باشگاه، هنوز هر پهلوان عضو یک باشگاه باشد، مساله پوشش مجموعه، پوششی با $k_{\text{set cover}}$ خواهد داشت. در این حالت چون باشگاه‌های تعطیل نشده، همه‌ی پهلوانان را پوشش می‌دهند پس همه‌ی مجموعه U به کمک $k_{\text{set cover}}$ عضو قابل پوشش خواهد بود.

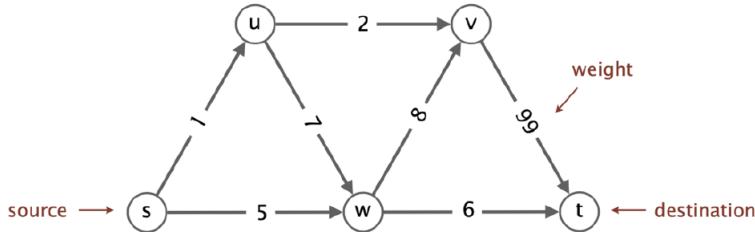
ثبت شد که این مساله هم NP و هم NP-Complete است پس NP-Hard نیز است.

(5) دو مساله یافتن کوتاهترین مسیر در گراف را در نظر بگیرید: (25 نمره)

مساله‌ی A: گراف وزن‌دار و جهتدار G با وزن‌های غیرمنفی و دوراس مبدا s و مقصد t داده شده است. کوتاهترین مسیر از s به t را پیدا کنید.

مساله‌ی B: گراف وزن‌دار و جهتدار G با وزن‌های غیرمنفی و دوراس مبدا s و مقصد t داده شده است. کوتاهترین مسیر از s به t را پیدا کنید اگر بتوانید از یال‌های این مسیر با وزن صفر عبور کنید؛ به عبارتی، وزن هر مسیر برابر با مجموع وزن یال‌های آن منهای وزن سنگین‌ترین یال است.

به عنوان مثال در گراف زیر کوتاهترین مسیر در مساله‌ی A برابر با $s \rightarrow w \rightarrow t$ است که وزن 11 دارد و در مساله‌ی B مسیر $s \rightarrow u \rightarrow v \rightarrow t$ است که وزن 3 دارد.



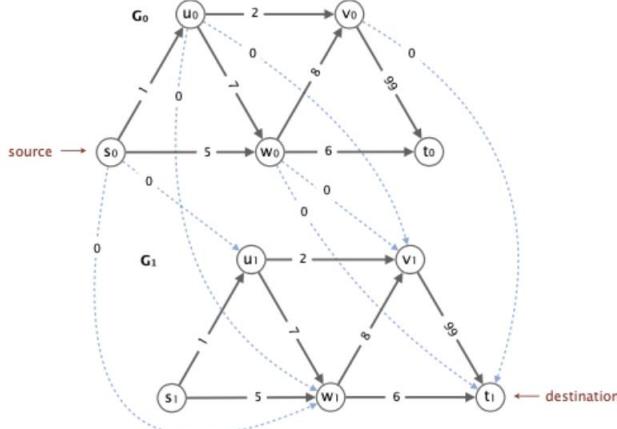
الف) یک reduction با زمان خطی از مساله‌ی A به B ارائه دهید.

ب) یک reduction با زمان خطی از مساله‌ی B به A ارائه دهید.

پاسخ:

الف) از گراف داده شده‌ی G یک گراف جدید G' می‌سازیم بدین صورت که یک راس جدید به نام s' به G اضافه کرده و یال s' را با وزن 1 $+ \max\{e \in E : w_e\}$ به آن می‌افزاییم. حال ادعا می‌کنیم کوتاهترین مسیری که مساله‌ی B برای رفتن از s' به t پیدا می‌کند حتماً از یال اضافه شده عبور می‌کند و شامل کوتاهترین مسیر از S به t نیز می‌باشد. صحبت این ادعا با توجه به نحوه ساختن G' واضح است زیرا تنها یالی که s' را به بقیه گراف متصل می‌کند یال اضافه شده است که سنگین‌ترین یال نیز هست و حذف خواهد شد و در نتیجه باقی یال‌ها از S به t مانند مساله‌ی A در نظر گرفته خواهد شد؛ اگر بخواهیم کوتاهترین مسیر برای G' را به کمک مساله A پیدا کنیم، می‌توانیم کوتاهترین مسیر از S به t را یافته و یال s' را به آن بیفزاییم که چون سنگین‌ترین یال است در انتها در نظر گرفته نخواهد شد؛ پس مساله‌ی A قابل تقلیل به مساله‌ی B در زمان خطی است.

ب) فرض کنید گراف ورودی داده شده G است. یک گراف جدید G' به کمک دو کپی G_0 و G_1 از G می‌سازیم و رؤوس اولی را با اندیس 0 و دومی را با اندیس 1 نمایش می‌دهیم. اگر یال $w \rightarrow v$ در گراف G وجود داشته باشد، یک یال با وزن صفر از v_0 به w_1 در گراف G' اضافه می‌کنیم. حال ادعا می‌کنیم برای یافتن کوتاهترین مسیر از S به t در گراف G (مساله‌ی B) می‌توان در مساله‌ی A کوتاهترین مسیر از s_0 به t_1 را در گراف G' پیدا کرد. وقتی کنید که برای این منظور، وقتی کوتاهترین مسیر در مساله‌ی B پیدا شد، مسیری که از گراف G_0 به گراف G_1 می‌رود معادل یالی است که الگوریتم B از آن با وزن صفر عبور می‌کند. باقی مسیر در گراف G_1 طی می‌شود که معادل آن در G_0 وجود دارد. به طور مشابه، کوتاهترین مسیر در گراف G (در مساله‌ی B) را می‌توان به جوابی در مساله‌ی A تبدیل کرد بدین صورت که یالی که از آن با وزن صفر عبور کرده‌ایم یالی خواهد بود که از G_0 به G_1 می‌رویم و باقی مسیر را به طور منتظر در G_1 طی می‌کنیم. پس مساله‌ی B قابل تقلیل به مساله‌ی A در زمان خطی است.



6) مجموعه A مجموعه‌ای با تعداد اعضای محدود است. مجموعه $Z = \{Z_1, Z_2, \dots, Z_m\}$ نیز وجود دارد به طوری که هر Z_i را تعدادی از اعضای A تشکیل می‌دهند و درواقع هر Z_i زیرمجموعه‌ای از A است. می‌خواهیم مجموعه A را با دو رنگ قرمز و سبز به گونه‌ای رنگ کنیم که در تمام اعضای هیچ Z_i ای همنگ نباشند. نشان دهید این مساله در دسته NP-Complete قرار دارد. (20 نمره)

پاسخ:

مساله‌ی صورت سوال را Opposite-Color می‌نامیم. ابتدا ثابت می‌کنیم این مساله در دسته‌ی NP قرار دارد؛ برای اینکار یک verifier یارانه می‌دهیم که به ازای تمامی Z_i ها چک می‌کند که تمامی اعضای آن همنگ نباشند؛ بدیهی است اینکار در زمان چندجمله‌ای امکان پذیر است پس مساله‌ی فوق در کلاس NP است.

برای اثبات NP-Hard بودن 3-SAT، مساله‌ی Opposite-Color 3-SAT که یک مساله‌ی NP-Complete است را به آن کاهش می‌دهیم. همانطور که می‌دانیم ورودی مساله 3-SAT یک 3-cnf است و متغیرهای آن x_1, x_2, \dots, x_n است. برای تبدیل این ورودی به ورودی مساله‌ی Opposite-Color، مجموعه‌ی A را به صورت زیر تعریف می‌کنیم: $A = \{x_1, \bar{x}_1, x_2, \bar{x}_2, \dots, x_n, \bar{x}_n, y\}$. در واقع A شامل تمام متغیرهای موجود در Φ به همراه نقیض آنها و متغیر y است؛ حال به ازای هر clause در Φ ، Z_i را برابر با متغیرهای موجود در آن clause به همراه متغیر y در نظر می‌گیریم؛ برای مثال به ازای $Z_i = \{x_1, \bar{x}_1, x_3, x_8, y\}$ مجموعه $C_i = \bar{x}_1 \vee x_3 \vee x_8$ را تشکیل می‌دهیم. همچنین مجموعه‌ی Z علاوه بر Z_i ها شامل n مجموعه $\{x_i, \bar{x}_i\}$ است. حال اگر Φ یک فرمول satisfiable باشد، رنگ‌آمیزی literal ها با دو رنگ، همان اختصاص true یا false به آنها است. اگر تمامی مقادیر true را با سبز و مقادیر false و متغیر y را با قرمز رنگ کنیم، هر یک از اعضای Z حداقل شامل یک قرمز هستند چون می‌دانیم y قرمز است و همچنین در مجموعه‌هایی که به صورت $\{x_i, \bar{x}_i\}$ تعریف شده بوند نیز قطعاً یک عضو قرمز و یک عضو سبز وجود دارد؛ هر یک از اعضای Z شامل حداقل یک عضو سبز نیز هست که clause متناظر با آن را true می‌کند؛ پس اگر در accept 3-SAT accept شود در accept Opposite-Color نیز accept می‌شود و در صورتی که در accept 3-SAT شود در accept 3-SAT نیز accept می‌شود چون قطعاً یک سبز در هر عضو مجموعه‌ی Z وجود داشته پس حداقل یکی از متغیرهایی که در 3-SAT باهم or شده‌اند true بوده است و این رنگ‌آمیزی معادل با یک satisfying assignment است. با اینکار در زمان چندجمله‌ای مساله‌ی Opposite-Color را به 3-SAT کاهش دادیم پس NP-Hard است و در پاراگراف اول اثبات شد NP است پس در دسته‌ی NP-Complete قرار دارد.

(7) با توجه به اینکه می‌دانیم مساله‌ی A، ثابت کنید مساله‌ی B در دسته مسائل NP-hard قرار دارد. (10 نمره امتیازی)

مساله‌ی A: تعیین اینکه آیا می‌توان مجموعه‌ای از اعداد را به دو گروه تقسیم کرد به طوری که جمع دو گروه باهم برابر شود.

مساله‌ی B: تعیین اینکه آیا می‌توان با کنار هم قرار دادن کاشی‌های مستطیلی در کف مستطیل شکل یک اتاق، تمام مساحت زمین را پوشاند بدون اینکه نیاز باشد کاشی‌ها را بشکنیم؟ (طول و عرض کاشی‌ها عدد طبیعی است)

پاسخ:

برای اینکه اثبات کنیم مساله‌ی B در دسته‌ی NP-Hard قرار دارد باید مساله‌ی A که در دسته‌ی NP-Complete قرار دارد را در زمان چندجمله‌ای به آن کاهش دهیم؛ اگر مجموعه وردی مساله A را برابر با $\{x_1, x_2, x_3, \dots, x_n\}$ در نظر بگیریم، برای تبدیل ورودی مساله‌ی A به ورودی مساله‌ی B، هر عدد x_i در مساله A را به یک کاشی $1 \times 4x_i$ تبدیل می‌کنیم و همچنین ابعاد کف اتاق را $y_i = \sum_{i=1}^n x_i$ در نظر می‌گیریم. دلیل در نظر گرفتن ضریب 4 در طول کاشی‌های این است که نتوانیم آن را در جهت عمودی در کف اتاق قرار دهیم. برای چیدن کاشی‌ها در کف اتاق باید آن را در دو ردیف بگنجانیم یعنی $\sum_{i=1}^n y_i = \frac{\sum_{i=1}^n 4x_i}{2}$ پس اگر مساله‌ی A جواب داشته باشد در مساله‌ی B نیز $\sum_{i=1}^n y_i = \sum_{i=1}^n 2x_i$ و مجموع هر دو ردیف باهم برابرند و اگر مساله‌ی B جواب داشته باشد، پس دو ردیف از کاشی‌ها چیده شده‌اند که مجموع طول آنها باهم برابر است. پس توانستیم در زمان چند جمله‌ای مساله‌ی A را به B کاهش دهیم پس B در دسته‌ی NP-Hard قرار دارد.



تاریخ تحويل: 1402/3/24



۱) درستی یا نادرستی هر یک از عبارات زیر را با ذکر دلیل مشخص کنید. (۱۲ نمره)

الف) اگر مساله B از کلاس مساله NP -Complete باشد و بدانیم $B \leq A$ آنگاه A در کلاس NP -Complete خواهد بود.

ب) اگر مساله A از کلاس NP -Complete باشد و بدانیم $A \leq B$ آنگاه B در کلاس NP -Complete خواهد بود.

ج) هر مساله‌ای NP -Complete را می‌توان در زمان چندجمله‌ای به هر مساله‌ای NP -Complete کاهش داد.

د) اگر داشته باشیم $P = NP$ آنگاه به ازای هر زبان X که در دسته P قرار دارد ($A, \sum \neq \emptyset$) در دسته NP -Complete نیز قرار خواهد داشت.

۲) فرض کنید a, b, c, p اعداد مثبت باینری هستند؛ ثابت کنید که $ModeP$ در دسته مسائل P قرار دارد. (۱۲ نمره)

$$(راهنمایی: (a^{1000})_2 = ((a^2)^2)^{1000})$$

$$ModeP = \{ \langle a, b, c, p \rangle \mid a^b \equiv c \pmod{p} \}$$

۳) یک دور "گذر دوبل از رئوس" در گراف بدون جهت G ، دوری است که از تمامی راس‌های G دقیقاً دوبار می‌گذرد. ثابت کنید مساله تشخیص وجود دور "گذر دوبل از رئوس" در گراف G ، یک مساله NP -Hard است. (۱۴ نمره)

۴) مساله‌ی مقابل را در نظر بگیرید: در شهر پهلوانان، هر پهلوان عضو حداقل یک باشگاه است. شهردار شهر برای افزایش تمرکز پهلوانان فصد دارد تعدادی باشگاه را تعطیل کند به طوری که پس از تعطیلی، هر پهلوان هنوز عضو حداقل یک باشگاه تعطیل نشده باشد. ورودی مساله، لیست پهلوانان، لیست باشگاه‌ها، لیست اعضای هر باشگاه و عدد k می‌باشد. آیا شهردار می‌تواند k باشگاه را طوری انتخاب کند که پس از تعطیلی آنها هنوز هر پهلوان عضو حداقل

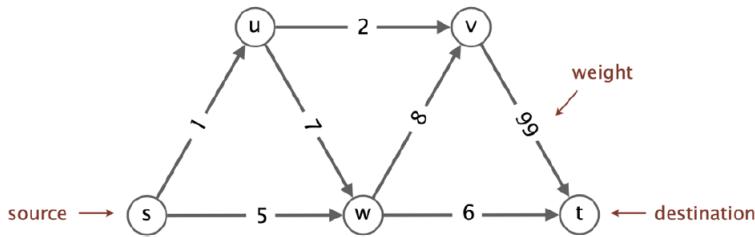
یک باشگاه باشد؟ ثابت کنید این مساله NP-Complete است. (راهنمایی: میتوانید از مساله [set cover](#) استفاده کنید.) (17 نمره)

(5) دو مساله یافتن کوتاهترین مسیر در گراف را در نظر بگیرید: (25 نمره)

مساله A: گراف وزن دار و جهت دار G با وزن های غیر منفی و دو راس مبدا S و مقصد t داده شده است. کوتاه ترین مسیر از S به t را پیدا کنید.

مساله B: گراف وزن دار و جهت دار G با وزن های غیر منفی و دور اس مبدا S و مقصد t داده شده است. کوتاه ترین مسیر از S به t را پیدا کنید اگر بتوانید از یکی از یال های این مسیر با وزن صفر عبور کنید؛ به عبارتی، وزن هر مسیر برابر با مجموع وزن یال های آن منهای وزن سنگین ترین یال است.

به عنوان مثال در گراف زیر کوتاه ترین مسیر در مساله A برابر با $t \rightarrow w \rightarrow s \rightarrow u \rightarrow t$ است که وزن 11 دارد و در مساله B مسیر $t \rightarrow v \rightarrow u \rightarrow s \rightarrow v \rightarrow t$ است که وزن 3 دارد.



الف) یک reduction با زمان خطی از مساله A به B ارائه دهید.

ب) یک reduction با زمان خطی از مساله B به A ارائه دهید.

(6) مجموعه A مجموعه ای با تعداد اعضای محدود است. مجموعه $Z = \{Z_1, Z_2, \dots, Z_m\}$ نیز وجود دارد به طوری که هر Z_i را تعدادی از اعضای A تشکیل می دهد و در واقع هر Z_i زیر مجموعه ای از A است. می خواهیم مجموعه A را با دو رنگ قرمز و سبز به گونه ای رنگ کنیم که در تمام اعضای هیچ Z_i ای هم رنگ نباشند. نشان دهید این مساله در دسته NP-Complete قرار دارد. (20 نمره)

(7) با توجه به اینکه می دانیم مساله A، NP-Complete است، ثابت کنید مساله B در دسته مسائل NP-hard قرار دارد. (10 نمره امتیازی)

مساله A: تعیین اینکه آیا می توان مجموعه ای از اعداد را به دو گروه تقسیم کرد به طوری که جمع دو گروه باهم برابر شود.

مساله B: تعیین اینکه آیا می توان با کنار هم قرار دادن کاشی های مستطیلی در کف مستطیل شکل یک اتاق، تمام مساحت زمین را پوشاند بدون اینکه نیاز باشد کاشی ها را بشکنیم؟ (طول و عرض کاشی ها عدد طبیعی است)