٨١٠١٠١۴٩٢ علیرضا کرمی

1)

a) ① demon picks: $p \geq 1$

② You pick: $w = a^p b^p a^p$, $|w| \geq p$

③ demon picks: $w = uvwxy$; $|vwx| \leq p$, $vx \neq \varepsilon$

حال برای رشته vwx سه حالت ممکن است رخ دهد، یا اینکه در قسمت a های بیفتد، یا اینکه در قسمت b بیفتد یا اینکه چون حداکثر طولش p است، یا به صورت $a^l b^j$ و یا به صورت $b^l a^j$ خواهد بود. پس برای مثال v می‌تواند به صورت a^k یا b^k یا $a^m b^n$ باشد. در حالت های دیگر به صورت a^k یا b^n است:

④ $i = 2 \longrightarrow uv^2wx^2y \longrightarrow$ تعداد b ها (حداقل) بیشتر از یکی از a ها می‌شود

یا تعداد a ها ناموازن می‌شود

برای حالتی که به صورت $a^m b^n$ است، اگر $i>1$ را بگیریم (برای مثال $i=2$)، فرمت رشته به صورت $a*b*a*$ نخواهد بود و قطعاً در زبان جای نخواهد گرفت. (کلاً اگر b اضافه شود، چون فقط یکی از a ها تغییر کرده، توان b از توان آن a ی دیگر بیشتر خواهد بود که خلاف قاعده زبان است)

b) ① demon picks ; $p \geq 1$

② you pick : $a^q b^q$; $q \geq p$; q is prime number

③ demon picks: $uvwxy$ $|vwx| \leq p$, $vx \neq \varepsilon$

④ you pick $i = q+1$

v و x مثل سوال قبل نمی‌توانند به شکل $a^m b^n$ باشند، چون با انتخاب $i>1$، فرمت زبان را نخواهند داشت

حال یا هر دو در یک بخش هستند (هر دو در a یا هر دو در b) یا v در a ها، x در b ها ست. برای هر کدام داریم:

1) هر دو در یک بخش باشند $\Longrightarrow i = q+1$ و $v = a^l$, $x = a^j$

$$\Longrightarrow w' = uv^{q+1}wx^{q+1}y = a^{q+lq+jq} = a^{q(1+l+j)}$$

چون $l + j > 0 \Longrightarrow$ توان a حداقل دارای دو فاکتور است پس عدد اول نیست و جزو زبان نیست.

2) v در a ها و x در b ها باشد $\Longrightarrow vx \neq \varepsilon \Longrightarrow$ توان v یا توان x صفر نیست.

$v = a^l$, $x = b^j$

بدون کاستن از کلیت فرض می‌کنیم توان v صفر نیست. داریم:

$$i = q+1 \Longrightarrow a^{q+lq} = a^{q(1+l)}$$

$\xRightarrow{l \neq 0}$ توان a دارای دو فاکتور است $\Rightarrow$ عدد اول نیست $\Rightarrow$ جزو زبان نیست

c) ① demon picks : $p \geq 1$

② You pick : $a^p b^p b^p a^p$

③ demon picks : $uvwxy$; $|vwx| \leq p$, $vx \neq \varepsilon$

④

رشته vwx حداکثر دارای طول p است، بنابراین حداکثر می‌تواند شامل دو بخش شود. یا بصورت a^*b^* یا b^*a^* یا a^* یا بصورت b^*. در هر حالت بررسی می‌کنیم

منظور یعنی بدین فرمت است و نه اینکه بی‌نهایت a داریم

ح1) اگر v بصورت a^l باشد: تعداد a ها در دو طرف بهم می‌خورد و دیگر $w \neq w^R$ $\quad i=2 \longrightarrow$

ح2) اگر v بصورت b^l و x بصورت b^j باشد: $\quad i=2 \longrightarrow n_b(w) > n_a(w)$

ح3) " " " " " " a^j باشد: $\quad i=2 \longrightarrow w \neq w^R$

ح4) اگر v بصورت $a^m b^n$ باشد (x قطعا بصورت b^j است) $\Longleftarrow$

چون از یک طرف رشته ما $\dots a^m b^n a^m b^n$ خواهیم داشت و از آن طرف دیگر نه، $w \neq w^R$ (یا x بصورت $b^n a^m$ باشد، دوباره همینطور می‌شود.) $\quad i=2 \longrightarrow w \neq w^R$

d) ① demon picks : $p \geq 1$

② you pick : $a^p b^p a^p b^p a^p b^p$

③ demon picks: $uvwxy$; $|vwx| \leq p$, $vx \neq \varepsilon$

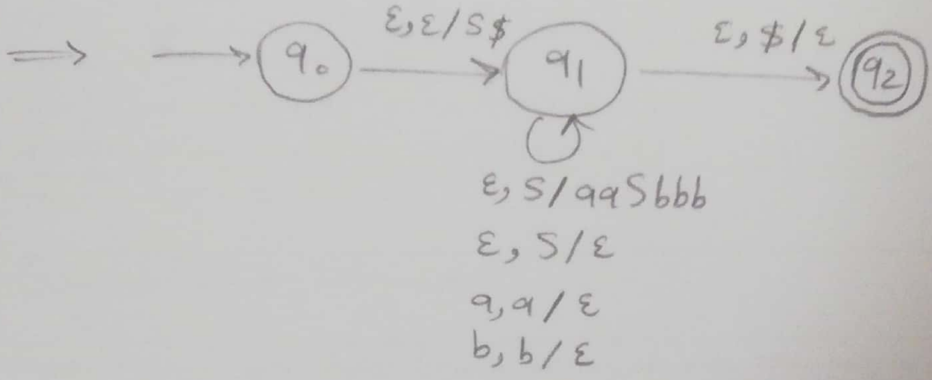
④ You pick $i=0 \longrightarrow$ توان‌های حداکثر یک a و حداکثر یک b کم می‌شود

و توازن توان‌ها بهم می‌خورد و دیگر نمی‌توانند سه بخش برابر داشته باشند.

(2

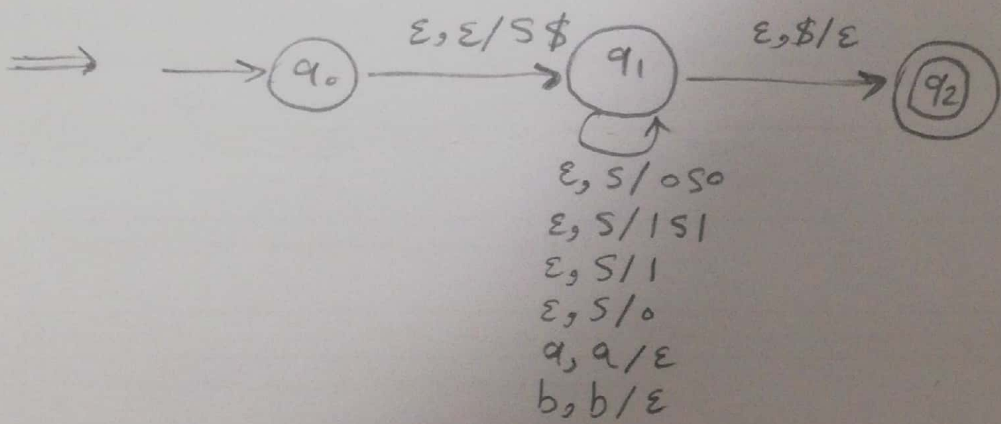
a) ابتدا گرامر CFL آن را می‌نویسیم و بعد آن را به PDA تبدیل می‌کنیم:

$S \rightarrow aaSbbb \mid \varepsilon$

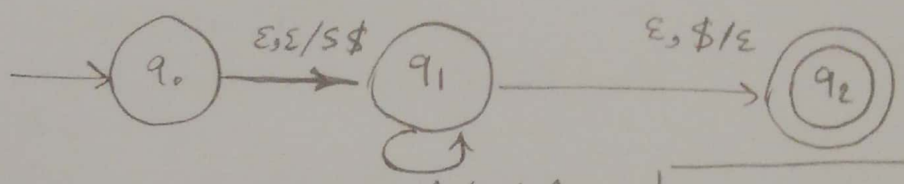


b) ابتدا گرامر CFL آن را می‌نویسیم و بعد آن را به PDA تبدیل می‌کنیم:

$S \rightarrow 0S0 \mid 1S1 \mid 0 \mid 1$



3) اگر بخواهیم گرامر این زبان ما به PDA تبدیل کنیم، خواهیم داشت:



در این حالت اگر دو قانون داشته باشیم بصورت

$A \to aA_1A_2 \dots$

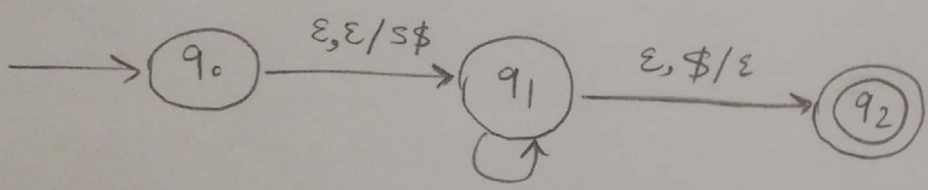
$A \to \beta A_1'A_2' \dots$

آنگاه باید دو قاعده transition به این صورت داشته باشیم:

$\varepsilon, A / aA_1A_2 \dots$

$\varepsilon, A / bA_1'A_2'' \dots$ $\Rightarrow$ deterministic نیست

حال از آنجایی که هر زوج (A, a) حداکثر یک بار بین قواعد دیده می‌شود، می‌توان PDA را به این صورت بازنویسی کرد:



$\Rightarrow$ DPDA

در واقع قواعد را به این صورت بازنویسی کردیم:

$$\left.\begin{array}{l} \varepsilon, A / aA_1A_2 \dots \\ a, a / \varepsilon \end{array}\right\} a, A / A_1A_2 \dots$$

اگر بیشتر از یک بار زوج مرتب (a, A) داشتیم، مثلا داشتیم:

$A \to aA_1A_2 \dots$

$A \to aA_1'A_2' \dots$

در مرحله ادغام و تبدیل خواهیم داشت:

$a, A / A_1A_2 \dots$

$a, A / A_1'A_2' \dots$

$\Leftarrow$ در این حالت deterministic نخواهد بود

۴)

چهار زبان زیر را تعریف می‌کنیم و گرامر آنها را می‌نویسیم

$$L_4 = \{a^n b^n \mid n = 4 \bmod 5\}$$

$$\xrightarrow{\text{گرامر}} \quad S \to aaaa\,X\,bbbb$$
$$X \to aaaaa\,X\,bbbbb \mid \varepsilon$$

$$L_2 = \{a^n b^n \mid n = 2 \bmod 5\}$$

$$\xrightarrow{\text{گرامر}} \quad S \to aa\,X\,bb$$
$$X \to aaaaa\,X\,bbbbb \mid \varepsilon$$

$$L_3 = \{a^n b^n \mid n = 3 \bmod 5\}$$

$$\xrightarrow{\text{گرامر}} \quad S \to aaa\,X\,bbb$$
$$X \to aaaaa\,X\,bbbbb \mid \varepsilon$$

$$L_1 = \{a^n b^n \mid n = 1 \bmod 5\}$$

$$\xrightarrow{\text{گرامر}} \quad S \to a\,X\,b$$
$$X \to aaaaa\,X\,bbbbb \mid \varepsilon$$

از آنجا که در صورت سوال خواسته شده که زبان نباید به فرمت $a^{5k} b^{5k}$ باشد، زبان مطلوب، اجتماع زبان‌های L_1 و L_2 و L_3 و L_4 است.

$$L = L_1 \cup L_2 \cup L_3 \cup L_4$$

می‌دانیم که زبان‌های مستقل از متن، تحت عملیات union، بسته هستند $\longleftarrow$ L نیز مستقل از متن است.

5) ابتدا ثابت می‌کنیم که $R \cap C$ مستقل از متن قطعی است. بعد ثابت خواهیم کرد که زبان‌های مستقل از متن قطعی، تنها عملیات "مکمل" بسته هستند. از آنها نتیجه می‌گیریم به کمک قانون دمورگان که $R \cup C$ نیز مستقل از متن قطعی است.

مرحله اول: ثابت می‌کنیم که اشتراک زبان مستقل از متن قطعی C و زبان منظم R، زبانی مستقل از متن قطعی خواهد بود.

فرض کنید که داریم: $DPDA = \{Q_1, \Sigma, \Gamma, \delta_1, q_{01}, F_1\}$ و $DFA = \{Q_2, \Sigma, \delta_2, q_{02}, F_2\}$. حال زبان اشتراک این دو زبان به این صورت خواهد بود:

$$DPDA' = \{Q, \Sigma, \Gamma, \delta, q_0, F\}$$

به طوریکه:

$$Q = Q_1 \times Q_2 \qquad \Sigma = \Sigma,\ \Gamma = \Gamma$$

$$q_0 = q_{01} \times q_{02}$$

در واقع هر زوج مرتب از state ها، یک استیت جدید می‌شوند

$$F = F_1 \times F_2$$

برای transition ها هم خواهیم داشت:

$$\delta = Q \times \Sigma \times \Gamma \qquad \delta(\overbrace{(q_1, q_2)}^{q}, a, b) = q' = (q_1', q_2')$$

$$q_1' = \delta_1(q_1, a, b), \quad q_2' = \delta_2(q_2, a)$$

$\Leftarrow$ ما در این شکل حالت هر دو ماشین را با یک زوج مرتب نشان می‌دهیم $(q = (\overset{DPDA}{q_1}, \overset{DFA}{q_2}))$

(در اشتراک DFA ها هم از همین ایده زوج مرتب استفاده می‌کردیم)

کتاب سیپسر

مرحله دوم: ثابت می‌کنیم که زبان مستقل از متن قطعی، تحت عملیات مکمل بسته خواهد بود. (اثبات از قضیه‌ای در کتاب سیپسر)

ابتدا هر استیت pop و read را به دو استیت فقط pop و فقط read تقسیم می‌کنیم و اگر آن state ها accept بودند، استیتی که read می‌کند را accept می‌کنیم. پس در DPDA ما تنها بعضی از استیت‌هایی که read می‌کنند accept هستند.

(در واقع اگر داشته باشیم $\delta(\underset{\text{استیت فعلی}}{q}, \underset{read}{a}, \underset{pop}{x}) = (\underset{\text{next state}}{r}, \underset{push}{y})$، حال خواهیم داشت: $\left\{ \begin{array}{l} \delta(q, \varepsilon, x) = (q_x, \varepsilon) \\ (q_x, a, \varepsilon) = (r, y) \end{array} \right.$)

$\Leftarrow$ زبان ما هنوز DPDA باقی مانده چون هیچ transition ی که non-deterministic باشد، اضافه نشده است.

حال کافی است برای مکمل گیری، استیت‌های read را برعکس کنیم. یعنی اگر accept بودند reject می‌شوند و اگر reject بودند، accept می‌شوند $\leftarrow$ هنوز DPDA است چون δ ها عوض نشدند و فقط F ها عوض شدند.

(مثل همان ایده مکمل گیری از زبان‌های منظم)

این اثبات طبق قضایای 2.41 و 2.42 کتاب سیپسر بیان شد.

ادامه به صفحه بعد

ادامه سوال 5) حال می دانیم طبق قانون دمورگان : $R \cup C = \overline{\overline{R} \cap \overline{C}}$

زبان های منظم و زبان های مستقل از متن قطعی، طبق اثبات صفحه قبل، تحت عملیات "مکمل" بسته هستند.

$\Longleftarrow$ $\overline{R}$ زبانی منظم و $\overline{C}$ زبانی مستقل از متن است.

حال اشتراک یک زبان منظم و یک زبان مستقل از متن قطعی، زبان مستقل از متن قطعی است. (اثبات صفحه قبل)

$\Longleftarrow$ $\overline{R} \cap \overline{C}$ زبانی مستقل از متن است.

قطعی

حال طبق قضایای 2.41 و 2.42 سیپسر و اثبات صفحه قبل، مکمل یک DCFL، یک DCFL است.

$\Longleftarrow$ $RUC = \overline{\overline{R} \cap \overline{C}}$ نیز زبانی مستقل از متن است.

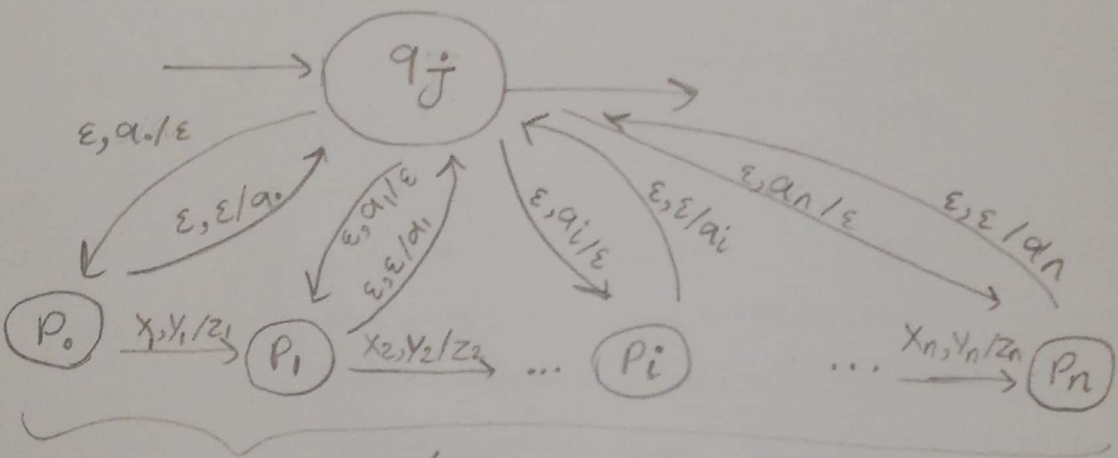
(۶

الف)

ابتدا سعی می‌کنیم برای راحتی در توضیح حل سوال، PDA و DFA را به گونه‌ای تغییر دهیم که یک استیت accept داشته باشند (هرکدام فقط) (مثلا می‌توان PDA را به simplified PDA تبدیل کرد و برای DFA هم از تمام استیت‌های accept، یک ε-transition می‌زنیم به یک استیت جدید و آن را accept می‌کنیم)

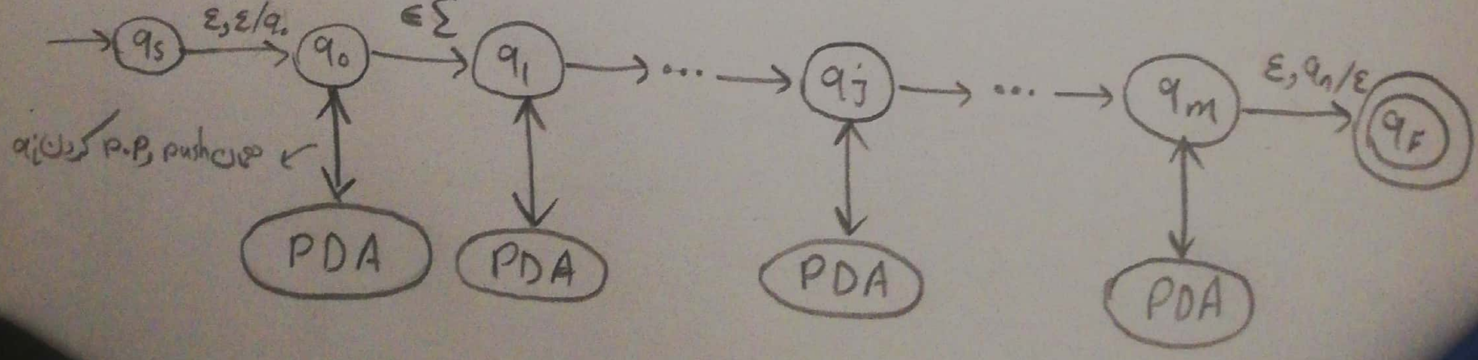
حال فرض کنید استیت‌های PDA را به صورت $p_0, p_1, \dots, p_n$ نوشتیم که p_n، accept است و استیت‌های DFA را به صورت $q_0, q_1, \dots, q_m$ نوشتیم که q_m، استیت accept آن DFA است.

حال به ازای هر استیت DFA این عملیات را انجام می‌دهیم:



کل PDA

در واقع هرگاه خواستیم از DFA به PDA برویم و زبان آن را بخوانیم حرف a_i را از سر استک pop می‌کنیم و هرگاه خواستیم از PDA به DFA برویم، حرف a_i را به سر استک اضافه می‌کنیم (push). این کار را برای این می‌کنیم که همیشه وقتی در DFA هستیم، بدانیم که تا کدام استیت از PDA را جلو رفته‌ایم و آن را ذخیره کرده باشیم و دفعه بعد که خواستیم از DFA به PDA برویم، از همان استیتی که تا آنجا جلو رفته بودیم ادامه دهیم. حال PDA کلی به این صورت خواهد شد:



توضیح صفحه بعد

ادامه قسمت ۶) الف)

در ابتدا برای PDA استیت p_n و برای DFA استیت q_m، accept بود. حال اگر ما به استیت p_n رسیده باشیم، روی استک a_n خواهیم داشت. تنها در صورتی که در استیت q_m باشیم (یعنی DFA زبان را گرفته است) و a_n هم سر استک باشد (یعنی زبان PDA هم گرفته شده است) آنگاه می‌توانیم به یک استیت accept برویم، با pop کردن a_n در استیت q_m.

قسمت ب) فرض کنید داشته باشیم: $L_1 = 0^n 2^n$, $L_2 = 1^m 3^m$. حال داریم:

$$L = \langle L_1 , L_2 \rangle \longrightarrow 0^n 1^m 2^n 3^m \in L$$

حال با استفاده از Pumping lemma ثابت می‌کنیم که $0^n 1^m 2^n 3^m$ مستقل از متن نیست.

1. demon picks : $p \geq 1$
2. you pick : $w = 0^p 1^p 2^p 3^p$
3. demon picks : $uvwxy$; $|vwx| \leq p$, $vx \neq \varepsilon$
4. You pick $i = 0$: $uv^0wx^0y \notin L$

توضیح: از آنجایی که طول vwx حداکثر p است، vwx حداکثر شامل یکی از 2 و 0 و یا 3 و 1 خواهد شد (مثلا می‌تواند بین 0 و 1، بین 1 و 2 یا بین 2 و 3 باشد و هم زمان نمی‌تواند هم شامل 0 هم شامل 2 باشد یا هم زمان هم شامل 1 هم شامل 3 باشد) بدون کاستن از کلیت فرض می‌کنیم شامل 0 است و یا 1 توان 0 کاهش می‌یابد اما توان 2 کاهش پیدا نمی‌کند و توان‌های 0 و 2 نابرابر می‌شود

$\Longleftarrow$ نمی‌تواند عضو L باشد.

مثال نقض

7) زبان L_1 را اینگونه درنظر می‌گیریم $L_1 = \{a^n b^m c \mid n \neq m,\ n,m \geq 1\}$

می‌دانیم که L_1 مستقل از متن است (کافی است به کمک استک کاری کنیم که تعداد a و b ها برابر نشوند و در آخر هم یک c به آن اضافه کنیم) و CFL آن هم به صورت روبرو است

$$S \to X_1 c \mid X_2 c$$

$$X_1 \to a X_1 \mid a X_1 b \mid \varepsilon$$

حال زبان L_2 را درنظر بگیرید: $L_2 = \{a^n b^m c^m \mid n,m \geq 1\}$

زبان L_2 نیز مستقل از متن است (کافی است a های ابتدایی را ignore کنیم، با b ها X را Push کنیم و با c ها X را Pop کنیم؛ اگر استک خالی شد، مورد قبول است.)

حال زبان $L = L_1 \cup L_2$ را درنظر می‌گیریم. چون CFL ها تحت اجتماع بسته هستند، زبان L نیز مستقل از متن خواهد بود. حال بررسی می‌کنیم که no_Prefix(L) نیز مستقل از متن است یا خیر. پس باید رشته‌هایی از L را پیدا کنیم که در no_prefix(L) حضور دارند. یعنی پیشوند آن رشته‌ها عضو زبان L نباشد. حال رشته‌های L_1 را بررسی می‌کنیم. رشته‌های L_1 چون به صورت $a^n b^m c$ هستند، پیشوندهای آن‌ها به صورت $a^n b^k$ یا a^k است که هیچ کدام از این دو حالت جزو زبان L نیست (نه در زبان L_1 نه در زبان L_2). پس تمام رشته‌های L_1 در no_Prefix(L) حضور دارند. حال رشته‌های زبان L_2 را بررسی می‌کنیم. اگر رشته‌های زبان L_2 به صورت $a^n b^m c^m$ باشند به طوری که $n \neq m$ است پیشوندهای آن به صورت $a^n b^m c^k$ یا $a^n b^k$ یا a^k است. از طرفی رشته‌هایی به صورت $a^n b^m c$ $(k=1)$ در L حضور دارند (این رشته‌ها از زبان L_1 آمده‌اند) و پیشوند آنها هستند. پس رشته‌های به صورت $a^n b^m c^m$ که $n \neq m$ است نمی‌توانند در no_Prefix(L) حضور داشته باشند. حال رشته‌های به فرم $a^m b^m c^m$ از زبان L_2 می‌مانند (حالتی که $n=m$). در این حالت پیشوندهای رشته‌ها به صورت $a^m b^m c^k$ یا $a^m b^k$ یا a^k است که هیچ کدام جزو زبان L نیستند (نه L_1 و نه L_2). پس رشته‌های به فرم $a^m b^m c^m$ از زبان L_2 می‌توانند در No_Prefix(L) حضور داشته باشند. در نهایت رشته‌های موجود در No_Prefix(L) به صورت $\underset{n \neq m}{a^n b^m c} \cup a^l b^l c^l$ خواهد بود. اما می‌دانیم که $a^l b^l c^l$ مستقل از متن نیست

$\Longleftarrow$ No_Prefix(L) مستقل از متن نیست