بسمه تعالی

نظریه زبان‌ها و ماشین‌ها

پاسخ تکلیف سوم

۱. برای زبان‌های زیر، گرامر مستقل از متن بنویسید. (منظور از wR وارون رشته w است.)

$L1 = \{a^n b^m \mid n \leq m + 3\}$

گرامر مربوط به آن را در زیر مشاهده می کنید:

$S \rightarrow aaaG|aaG|aG|G$

$G \rightarrow aGb|B$

$B \rightarrow Bb|\epsilon$

متغیر شروع ما، در اول حداکثر سه عدد aرا سمت چپ Gاضافه می کند. خود Gهم تا زمانی که به Bتبدیل نشود، تعداد aو b برابری را در سمت چپ و راست رشته اضافه می کند. به عبارتی، mو n به صورت برابر زیاد می کند. در آخر هم، با تبدیل شدن به Bمقدار دلخواهی به bها اضافه می شود، چون Bتعداد دلخواهی bرا می تواند تولید کند. این به این معنی است که mرا هر چقدر که خواستیم می توانیم زیاد کنیم. طبق حالتی که این گرامر را ساختیم، می فهمیم که تمام رشته هایی که به فرم $a^n b^m$ هستند و $n \leq m + 3$را تولید می کنیم. چون مقداری که nمی تواند از mبیشتر باشد را در Sزیاد شدن با هم mو nرا در Gو زیاد شدن دلخواه mرا در Bهندل کرده ایم.

$L2 = \{a^n b^m c^k \mid k = |n - m|\}$

گرامر آن را در زیر مشاهده می کنید:

$S \rightarrow S+|GS-$

$S+ \rightarrow a\ S+\ c|G$

$S- \rightarrow b\ S-\ c$

$G \rightarrow a\ G\ b|\epsilon$

در این گرامر، در اول کار بین علامت عبارت n - mتصمیم گیری می شود. اگر مثبت بود، سراغ +S می رویم و اگر منفی بود سراغ -S در +Sبه تعدادی nبیشتر از mهست، aدر سمت چپ و cدرسمت راست رشته اضافه می کنیم. هر موقع تعداد لازم را ایجاد کردیم، متغیر را به Gتبدیل می کنیم. این متغیر صرفا تعداد برابری aو bدر سمت چپ و راست ایجاد می کند. پس در واقع مقدار $n - m$ را ثابت نگه می دارد. در -S هم، با استفاده از Gسمت چپ آن مقدار مینیمم مربوط به nو mراحرف aو bایجاد می کنیم. سپس، خود -S تعداد برابری bو cتولید می کند. این عمل باعث می شود که مقدار m - nتا حرف cداشته باشیم. همانطور که توضیح داده شد، این گرامر همیشه شرط سوال یعنی $nc = |na - nb|$ را برقرار نگه می دارد و در فرمت $a^n b^m c^k$ هم هست. پس زبانی که تشخیص می دهد دقیقا با زبان Lیکسان است.

$L3 = \{a^n b^m \mid 2n \le m \le 3n\}$

گرامر مربوط به آن را در زیر مشاهده می کنید:

$S \rightarrow aSbb|aSbbb|\epsilon$

همانطور که مشاهده می شود، متغیر شروع یا به رشته ی تهی تبدیل می شود، یا یک aبه سمت چپ رشته اضافه می کند. با اضافه کردن این aآن را متناظر با دو یا سه bاضافه می کند و این تعداد bرا هم به راست رشته اضافه می کند. این به این معنی است که هر حرف a در تناظر با دو یا سه حرف b است. پس می فهمیم که nbحداقل 2naو حداکثر 3na است. پس تمام رشته های تعریف Lرا تولید می کنیم و یک گرامر مستقل از متن برای این زبان داریم.

$L4 = \{w \mid w \in \{a,b,c\}^*, na(w) + nb(w) = nc(w) \}$

گرامر مربوط به آن را در زیر مشاهده می کنید:

$S \rightarrow \epsilon \mid aScS \mid cSaS \mid bScS \mid cSbS$

به ازای هر حرف c که قرار میدهیم یا باید یک حرف a بگذاریم یا یک حرف b و حالت های مختلف آن را در نظر میگیریم.

$L5 = \{w1\#w2 \mid w1, w2 \in \{a,b\}^*, w1 \neq w2R \}$

گرامر مربوط به آن را در زیر مشاهده می کنید:

$S \rightarrow aSa|aSb|bSa|bSb|T|Y$

$T \rightarrow aXb|bXa$

$X \rightarrow aXb|bXb|bXa|aXa|\#$

$Y \rightarrow ZX|XZ$

$Z \rightarrow aA|bB|a|b$

$L6 = \{anbm \mid n = m - 1\}$

گرامر مربوط به آن را در زیر مشاهده می کنید:

$S \rightarrow Gb$

$G \rightarrow aGb|\epsilon$

متغیر شروع این گرامر، یک bبه سمت راست رشته اضافه می کند و به متغیر Gتبدیل می شود. متغیر Gهم در هر مرحله، یک aو یک bبه چپ و راستش اضافه می کند. این به این معنی است که در مرحله ی nام، به راست رشته nتا bو به سمت چپ nتا aاضافه شده است. این ها روی هم به همراه bاولی که در Sایجاد شده بود، رشته های به فرم $a^n b^m$ با شرط $n = m - 1$را ایجاد می کند. این، همان زبان Lاست.

۲. بر روی گرامر زیر عملیات ساده‌سازی و حذف اپسیلون رول ها و یونیت رول ها را به ترتیب صحیح انجام دهید. (λ همان رشته‌ی تهی است)

$S \rightarrow ABED$
$E \rightarrow EaE|B$
$B \rightarrow bB|Bb|\lambda$
$A \rightarrow AgA|EE$
$D \rightarrow DA|c$

نکته در ابتدا باید $S0 \rightarrow S$ را اضافه کنید اما چون در اینجا تغییر در مساله ایجاد نمیکند میتوان اضافه نکرد گرچه برای افزایش دقت بهتر است اضافه شود.

ϵ rules: به ترتیب مراحل حذف این قوانین را را مشاهده می کنید:

$S \rightarrow ABED|AED$
$E \rightarrow EaE|B|\epsilon$
$B \rightarrow b|bB|Bb$
$A \rightarrow AgA|EE$
$D \rightarrow DA|c$

$S \rightarrow ABED|AED|ABD|AD$
$E \rightarrow EaE|Ea|aE|a|B$
$B \rightarrow b|bB|Bb$
$A \rightarrow AgA|E|EE|\epsilon$
$D \rightarrow DA|c$

دقت کنید $B \rightarrow b|bB|Bb$ با $B \rightarrow b|bB$ معادل است.

$S \rightarrow ABED|AED|ABD|AD|BED|ED|BD|D$
$E \rightarrow EaE|Ea|aE|a|B$
$B \rightarrow b|bB$
$A \rightarrow AgA|gA|Ag|g|E|EE$
$D \rightarrow D|DA|c$

unit rules: به ترتیب مراحل حذف این قوانین را را مشاهده می کنید:

$S \rightarrow ABED|AED|ABD|AD|BED|ED|BD|D$
$E \rightarrow EaE|Ea|aE|a|B$
$B \rightarrow b|bB$
$A \rightarrow AgA|gA|Ag|g|E|EE$
$D \rightarrow D|DA|c$

$S \rightarrow ABED|AED|ABD|AD|BED|ED|BD|D$
$E \rightarrow EaE|Ea|aE|a|b|bB$
$B \rightarrow b|bB$
$A \rightarrow AgA|gA|Ag|g|EaE|Ea|aE|a|B|EE$
$D \rightarrow DA|c$

$S \rightarrow ABED|AED|ABD|AD|BED|ED|BD|DA|c$
$E \rightarrow EaE|Ea|aE|a|b|bB$
$B \rightarrow b|bB$
$A \rightarrow AgA|gA|Ag|g|EaE|Ea|aE|a|b|bB|EE$
$D \rightarrow DA|c$

خواسته سوال تا همین مرحله است اما میتوان به فرم نرمال چامسکی نیز تبدیل کرد.

$S \rightarrow ZW|AW|ZD|AD|BW|ED|BD|DA|c$
$E \rightarrow TE|EX|XE|a|b|bB$
$B \rightarrow b|bB$
$A \rightarrow UA|YA|AY|g|TE|EX|XE|a|b|bB|EE$
$D \rightarrow DA|c$

$X \rightarrow a$

$Y \rightarrow g$

$Z \rightarrow AB$

$W \rightarrow ED$

$T \rightarrow EX$

$U \rightarrow AY$

۳. برای هر یک از موارد زیر مشخص کنید که کدام یک از گرامر های زیر مبهم هستند. توضیح دهید.

$S \rightarrow aAB$
$A \rightarrow bBb$
$B \rightarrow A \mid \epsilon$

مبهم است رشته abbbb را درنظر بگیرید از دو LMD میتوان آن را حاصل کرد.

$S \Rightarrow aAB \Rightarrow abBbB \Rightarrow$

$B \rightarrow A \quad abAbB \Rightarrow abbBbbB \Rightarrow abbbb$

$B \rightarrow \epsilon \quad abbB \Rightarrow abbA \Rightarrow abbbBb \Rightarrow abbbb$

$S \rightarrow SS \mid aSb \mid bSa \mid \epsilon$

مبهم است رشته abab را درنظر بگیرید از دو LMD میتوان آن را حاصل کرد.

$$S \Rightarrow SS \Rightarrow aSbS \Rightarrow abS \Rightarrow abaSb \Rightarrow \boxed{abab}$$
$$S \Rightarrow aSb \Rightarrow abSab \Rightarrow \boxed{abab}$$

S → AB | aaB
A → a | Aa
B → b

مبهم است رشته aab را درنظر بگیرید از دو LMD میتوان آن را حاصل کرد.

$$S \Rightarrow aaB \Rightarrow \boxed{aab}$$
$$S \Rightarrow AB \Rightarrow AaB \Rightarrow aaB \Rightarrow \boxed{aab}$$

۴. گرامر زیر را بر فرم نرمال چامسکی تبدیل کنید. (λ همان رشته‌ی تهی است)

S → baAB
A → bAB|λ
B → BAa|A|λ

مطابق با روند زیر برای این گرامر عمل می کنیم:

نکته در ابتدا باید S0 → S را اضافه کنید اما چون در اینجا تغییر در مساله ایجاد نمیکند میتوان اضافه نکرد گرچه برای افزایش دقت بهتر است اضافه شود.

حذف ϵ ruleها: شبیه روشی که سر کلاس ارائه شد، قوائد اپسیلون را حذف می کنیم. این روندرا در زیر مشاهده می کنید:

S → baAB
A → bAB|ϵ
B → BAa|A|ϵ

S → baAB|baB
A → bAB|bB
B → BAa|A|Ba|ϵ

S → baAB|baB|baA|ba
A → bAB|bB|bA|b
B → BAa|A|Ba|Aa|a

حذف unit ruleها: شبیه روشی که سر کلاس ارائه شد، قوائد یکه را حذف می کنیم. این روند را در زیر مشاهده می کنید:

$S \rightarrow baAB|baB|baA|ba$
$A \rightarrow bAB|bB|bA|b$
$B \rightarrow BAa|A|Ba|Aa|a$

$S \rightarrow baAB|baB|baA|ba$
$A \rightarrow bAB|bB|bA|b$
$B \rightarrow BAa|Ba|Aa|a|bAB|bB|bA|b$

درست کردن فرم قوائد: با استفاده از متغیر های اضافی، فرم قوائد را به قوائد چامسکی تبدیل می کنیم. برای این کار، متغیر های زیر را با این قوائد تعریف می کنیم. این متغیر ها را طوری انتخاب کرده ایم که نیازی به متغیر اضافه نباشد.

$A* \rightarrow a$
$B* \rightarrow b$
$C* \rightarrow B*A*$
$C \rightarrow AB$
$D \rightarrow AA*$

حال، با استفاده از این متغیر ها قوائد اصلی را به صورت زیر بازنویسی می کنیم:

$S \rightarrow C*\ C|C*\ B|C*\ A|B*\ A*$
$A \rightarrow B*\ C|B*\ B|B*\ A|b$
$B \rightarrow BD|B\ A*|A\ A*|a|B*\ C|B*\ B|B*\ A|b$

۵. ثابت کنید اگر L مستقل از متن باشد، Suffix(L) نیز مستقل از متن است.

$\text{Suffix}(L) = \{\ y \mid xy \in L\ \}$

ما می توانیم این ادعا را با نشان دادن وجود یک CFG که می تواند پسوندهای L را ایجاد کند، اثبات کنیم.

از آنجایی که L یک CFL است، پس یک گرامر بدون متن به شکل نرمال چامسکی برایش وجود دارد که با G نشان میدهیم.

همچنین یک CFG جدید تشکیل می دهیم که می تواند پسوند L را ایجاد کند و آن را G’ می‌نامیم. برای انجام این کار، ما G را به روش زیر تغییر دهیم:

(۱) برای هر متغیر X از G، یک متغیر X’ اضافه کنید. (X’ پسوندهای زبانی را ایجاد میکند که با استفاده از X به عنوان start variable تولید می شود.)

(۲) اگر X →YZ یک رول G باشد، دو رول جدید اضافه می کنیم، X’→ Z’ X’→ Y’Z

(۳) همچنین، برای هر متغیر در G اضافه می کنیم: X ’→ ϵ X ’→ X

(۴) اگر S متغیر شروع G باشد، S’ را متغیر شروع G' میگذاریم.

اکنون G’ جدید می‌تواند پسوند L را تولید کند، و از این رو زبان بدون متن تحت عملیات پسوند بسته است.