فصل ۶ ساده‌سازی گرامرهای مستقل از متن

همانطور که در فصل گذشته بیان شد، برخی گرامرهای مستقل از متن به دلیل داشتن قوانین نامطلوب از کارایی بالایی برخوردار نیستند. استفاده از گرامرهای مستقل از متن که سادگی لازم را نداشته باشند در طراحی کامپایلرها سرعت کار را بسیار پایین آورده و موجب کاهش کارایی می‌شود ولی استفاده از الگوریتم‌هایی جهت ساده‌سازی گرامرها می‌تواند بسیار در افزایش کارایی یک کامپایلر مؤثر باشد.

برای حذف قواعد نامطلوب بر اساس ترتیب زیر عمل می‌کنیم:

۱. حذف قوانین λ

۲. حذف قوانین یکه

۳. حذف قوانین بی‌فایده

الگوریتم حذف قوانین λ

قواعدی به فرم $A \to \lambda$ قانون پوچ یا قانون λ نامیده می‌شود. همچنین متغیری مانند A که در آن $A \overset{*}{\Rightarrow} \lambda$ متغیر میرا (nullable) نامیده می‌شود. برای حذف کردن قواعد پوچ ابتدا باید متغیرهای میرا را مشخص کرده و سپس در همه قوانین گرامر آنها را با λ جایگزین کرد. توسط الگوریتم زیر می‌توان متغیرهای میرا را در مجموعه U مشخص کرد:

$U \leftarrow \{\lambda\}$

Loop

If there exists a production rule $A \to x$ such that $x \in U^*$ **then**

Add A to U

Else

Exit Loop

End Loop

مثال۱: در گرامر زیر متغیرهای A، B، C و D میرا محسوب می‌شوند:

$S \to aAB \mid b$

$A \to BCD$

$B \to CD$

$C \to D$

$D \to \lambda$

حال با جایگزینی متغیرهای میرا با λ و حذف قوانین پوچ به گرامر زیر می‌رسیم:

$S \to aAb \mid b \mid aB \mid aA \mid a$

$A \to BCD \mid CD \mid BD \mid BC \mid D \mid C$

$B \to CD \mid C \mid D$

$C \to D$

> **نکته:** حذف قوانین پوچ در گرامرهایی امکان‌پذیر است که متغیر آغازین گرامر میرا نباشد به بیان دیگر $\lambda \notin L(G)$ و در غیر اینصورت می‌توان با الگوریتم بالا به گرامری دست یافت که زبان $L(G)-\{\lambda\}$ را تولید می‌کند.

الگوریتم حذف قوانین یکه

تعریف: گراف وابستگی برای گرامر داده شده G به این صورت است که برای هر متغیر گرامر رأسی در گراف وجود دارد و به ازای هر قانونی به فرم

$$\begin{cases} A \to xBy \\ A, B \in V \\ x, y \in (V \cup T)^* \end{cases}$$

از رأس متناظر با متغیر A یالی به رأس متناظر با رأس B در گراف وابستگی برقرار می‌گردد.

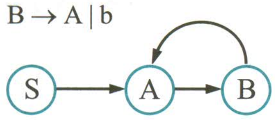
قواعدی به فرم $A \to B$ که در آن A و B متغیر می‌باشند قانون یکه نامیده می‌شود. برای حذف قوانین یکه ابتدا با ساختن یک گراف وابستگی مربوط به قوانین یکه، متغیرهایی که فقط با قوانین یکه می‌توانند یک متغیر دیگر را تولید کنند را شناسایی می‌کنیم. برای این کار به این صورت عمل می‌کنیم که در صورتی که در گراف وابستگی برای دو متغیر مانند A و B مسیری در آن گراف وجود داشته باشد می‌توان نتیجه گرفت که $A \overset{*}{\Rightarrow} B$ که در این صورت اگر در گرامر قوانینی به فرم $B \to y_1 \mid ... \mid y_n$ وجود داشته باشد قوانینی به فرم $A \to y_1 \mid ... \mid y_n$ را به گرامر می‌افزاییم و قوانین یکه را از آن حذف می‌کنیم.

مثال ۲: برای گرامر زیر ابتدا گراف وابستگی مربوط به قوانین یکه را تولید می‌کنیم:

$S \to aAB \mid A$

$A \to B \mid a$

$B \to A \mid b$



از روی گراف وابستگی نتیجه می‌شود که:

$1)\ S \overset{*}{\Rightarrow} A$

$2)\ S \overset{*}{\Rightarrow} B$

$3)\ A \overset{*}{\Rightarrow} B$

$4)\ B \overset{*}{\Rightarrow} A$

حال از روی قانون شماره ۱ می‌توان دریافت که باید قواعد زیر را به گرامر افزود: $S \to a$

از روی قانون شماره ۲ می‌توان دریافت که باید قواعد زیر را به گرامر افزود: $S \to b$

از روی قانون شماره ۳ می‌توان دریافت که باید قواعد زیر را به گرامر افزود: $A \to b$

و از روی قانون شماره ۴ می‌توان دریافت که باید قواعد زیر را به گرامر افزود:

$B \rightarrow a$

باید توجه داشت که در این کار قواعد یکه به گرامر افزوده نمی‌شوند و قواعد یکه موجود نیز حذف می‌شوند و لذا گرامر تغییریافته به صورت زیر خواهد شد:

$S \rightarrow aAB \mid a \mid b$

$A \rightarrow a \mid b$

$B \rightarrow b \mid a$

الگوریتم حذف قوانین بی‌فایده

تعریف: متغیری مانند A بافایده است اگر دارای دو شرط زیر باشد:

$$\begin{cases} S \overset{*}{\underset{(1)}{\Rightarrow}} xAy \overset{*}{\underset{(2)}{\Rightarrow}} w \\ w \in T^* \\ x, y \in (V \cup T)^* \end{cases}$$

به بیان دیگر متغیری بافایده است که بتواند در اشتقاق یک رشته ظاهر شود. هر متغیری که یکی از شروط ۱ یا ۲ را نداشته باشد متغیری بی‌فایده است و هر قانونی که شامل متغیر بی‌فایده باشد یک قانون بی‌فایده محسوب می‌شود. برای شناسایی متغیرهای بی‌فایده ابتدا متغیرهایی که شرط ۲ را ندارند را به کمک الگوریتم زیر شناسایی می‌کنیم:

```
U ← T
Loop
    If there exists a production rule A → x such that x ∈ U* then
        Add A to U
    Else
        Exit Loop
End Loop
```

متغیرهایی که پس از اتمام الگوریتم فوق در مجموعه U قرار ندارند آنهایی هستند که شرط ۲ را ندارند. بعد از شناسایی متغیرهای بی‌فایده که شرط ۲ را ندارند کلیه قوانین شامل آنها را از گرامر حذف می‌کنیم و سپس توسط تولید گراف وابستگی گرامر، رئوسی از گراف که از رأس متناظر با متغیر آغازین گرامر مسیری به آنها وجود ندارد را مشخص می‌کنیم. آنها متغیرهایی هستند که شرط ۱ را ندارند. با شناسایی آنها و حذف کلیه قوانین شامل آنها از گرامر،به گرامر بدون قوانین بی‌فایده خواهیم رسید.

مثال ۳ : گرامر زیر را در نظر بگیرید:

$S \rightarrow a \mid aA \mid bBb \mid aC$

$A \rightarrow aB \mid \lambda$

$B \rightarrow Aa$

$C \rightarrow cCD$

$D \rightarrow ddd$

$F \rightarrow EGa$

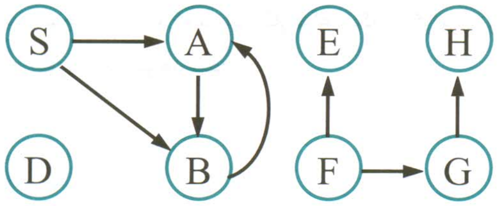
$G \rightarrow Hb$

$H \rightarrow a$

$E \rightarrow b$

در گرامر مذکور متغیر C شرط ۲ را ندارد و لذا کلیه قوانین شامل C از گرامر حذف می‌شود. سپس با رسم گراف وابستگی به

صورت زیر به این نتیجه می‌رسیم که متغیرهای D، H، E، G و F شرط ۱ را ندارند و لذا با حذف قوانین شامل آنها به گرامر زیر می‌رسیم:



$S \to a \mid aA \mid bBb$

$A \to aB \mid \lambda$

$B \to Aa$

پیچیدگی گرامرها پس از حذف قوانین نامطلوب

۱. با حذف قوانینی همچون $A \to \lambda$ و $A \to B$ تغییر مهمی در قدرت گرامر ایجاد نمی‌شود. منظور از ساده سازی گرامرها حذف بعضی از قوانین نامطلوب است که همیشه منجر به کاهش تعداد قوانین نمی‌شود. حذف قوانین بی‌فایده همیشه پیچیدگی را کاهش میدهد. اما الزاماً گرامر کمینه (بهینه) تولید نمی کند.

۲. در مورد روال حذف قانون λای مثل $A \to \lambda$ قاعده λ حذف می‌شود و سپس به ازای هر قاعده به صورت $B \to xAy$ قاعده $B \to xy$ به P اضافه می‌شود. در این حالت پیچیدگی زیاد خواهد شد. ولی در حالتی که قاعده λی $A \to \lambda$ وجود داشته باشد و متغیر A در سمت رأست هیچ قاعده‌ای نباشد فقط یک قاعده کم می‌شود پس پیچیدگی ثابت می‌ماند. بنابراین در حذف قوانین λ پیچیدگی یا افزایش می‌یابد و یا ثابت باقی می‌ماند.

۳. در مورد قوانین یکه برای قاعده یکه $A \to B$ ابتدا این قاعده را حذف و سپس به ازای هر قاعده به صورت $B \to x$ که در آن $x \in (V \cup T)^{\star}$ و همچنین x یک متغیر تنها نیست (با در نظر گرفتن این موضوع که گرامر از ابتدا شامل قانون λ نیست) قاعده $A \to x$ اضافه خواهد شد. در این جا بسته به تعداد قوانینی که به صورت $B \to x$ هستند و همچنین طول x، ممکن است پیچیدگی ثابت بماند یا اضافه شود. بنابراین در حذف قوانین یکه پیچیدگی یا افزایش می‌یابد و یا ثابت باقی می‌ماند.

نکاتی در رابطه با قوانین λ، یکه و بی‌فایده

۱. در حذف قواعد نامطلوب حتماً باید بر اساس ترتیب زیر عمل کرد:

i. حذف قوانین λ

ii. حذف قوانین یکه

iii. حذف قوانین بی‌فایده

۲. حذف قوانین λ ممکن است باعث ایجاد قوانین یکه یا بی‌فایده‌ای شود که قبلاً وجود نداشته است.

۳. حذف قوانین یکه ممکن است باعث ایجاد قوانین بی‌فایده‌ای شود که قبلاً در گرامر وجود نداشته است. ولی حذف قوانین یکه باعث ایجاد λ نمی‌شود.

۴. حذف قوانین بی فایده باعث ایجاد قوانین یکه یا λ نمی‌شود.

حذف چپ گردی و راست گردی

فرض کنید $G = (V, T, S, P)$ یک گرامر مستقل از متن باشد. مجموعه قوانینی که سمت چپ آن ها متغیری مانند A است را به دو زیر مجموعه جدای زیر تقسیم کنید:

$$A \to Ax_1 \mid Ax_2 \mid \dots \mid Ax_n$$

$$A \to y_1 \mid y_2 \mid \dots y_m$$

که در آن $x_i, y_i \in (V \cup T)^*$ است اما A پیشوندی از هیچ y_i نمی باشد. گرامر $\hat{G} = (V \cup \{Z\}, T, S, \hat{P})$ را ملاحظه کنید که

در آن $\hat{P}, Z \notin V$ با جایگزین کردن تمامی قوانینی که در سمت چپشان A دارند و بوسیله قوانین زیر ایجاد می شوند بدست می آید:

$$A \rightarrow y_i \mid y_i Z \quad : i = 1, 2, \ldots m$$

$$Z \rightarrow x_i \mid x_i Z \quad : i = 1, 2, \ldots n$$

آن‌گاه: $L(G) = L(\hat{G})$.

متمم چپ گردی، همان رأست گردی است که برای حذف رأست گردی به صورت زیر عمل می کنیم:

فرض کنید G=(V,T,S,P) یک گرامر مستقل از متن باشد. مجموعه قوانینی که متغیر A در سمت چپ آن ها است به دو زیر مجموعه جدا و مستقل زیر تقسیم می شوند:

$$A \rightarrow y_1 \mid y_2 \mid \ldots y_m$$

$$A \rightarrow x_1 A \mid x_2 A \mid \ldots \mid x_n A$$

که در آن A هیچ پیشوندی از y_i نیست. گرامری که از جایگزینی قوانین زیر بدست می آید معادل گرامر اصلی است.

$$A \rightarrow y_i \mid Z y_i \quad : i = 1, 2, \ldots m$$

$$Z \rightarrow x_i \mid Z x_i \quad : i = 1, 2, \ldots n$$

نکته: معمولاً در گرامرها چپ‌گردی نامطلوب محسوب می‌شود.

مثال ۴ : چپ گردی را از گرامر G حذف کنید:

$$G: \begin{array}{l} A \rightarrow Aa \mid aBc \mid \lambda \\ B \rightarrow Bb \mid bc \end{array}$$

حل: قاعده‌های $A \rightarrow Aa$, $B \rightarrow Bb$ دارای چپ گردی هستند:

$$A \rightarrow A\underbrace{a}_{x_1} \Big| \underbrace{aBc}_{y_1} \Big| \underbrace{\lambda}_{y_2} \Rightarrow \begin{cases} A \rightarrow aBc \mid \lambda \mid aBcZ_1 \mid Z_1 \\ Z_1 \rightarrow a \mid aZ_1 \end{cases}$$

$$B \rightarrow B\underbrace{b}_{x_2} \Big| \underbrace{bc}_{y_2} \Rightarrow \begin{cases} B \rightarrow bc \mid bcZ_2 \\ Z_2 \rightarrow b \mid bZ_2 \end{cases}$$

و در نهایت داریم:

$$G: \begin{array}{l} A \rightarrow aBc \mid \lambda \mid aBcZ_1 \mid Z_1 \\ Z_1 \rightarrow a \mid aZ_1 \\ B \rightarrow bc \mid bcZ_2 \\ Z_2 \rightarrow b \mid bZ_2 \end{array}$$

فرم نرمال چامسکی

تعریف: گرامر G در شکل نرمال چامسکی است اگرتمامی قواعد آن به صورت زیر باشد:

$$A \rightarrow BC \text{ or } A \rightarrow a$$

$$\begin{cases} A, B, C \in V \\ a \in T \end{cases}$$

فرم نرمال گریباخ

تعریف: گرامر G در شکل نرمال گریباخ است اگر تمامی قواعد آن به صورت زیر باشد:

$$A \to ax$$

$$\begin{cases} A \in V \\ a \in T \\ x \in V^* \end{cases}$$

نکته: هر گرامر مستقل از متن $G=(V.T,S,P)$ که در آن $\lambda \notin L(G)$ باشد دارای گرامر معادلی در فرم نرمال چامسکی است.

نکته: هر گرامر مستقل از متن $G=(V.T,S,P)$ که در آن $\lambda \notin L(G)$ باشد دارای گرامر معادلی در فرم نرمال گریباخ است.

مثال ۵: گرامر زیر را در نظر بگیرید:

$$S \to abCD \mid b$$
$$C \to cd$$
$$D \to d$$

ابتدا باید قواعد نامطلوب را از گرامر حذف کرده و سپس با تغییر و جایگزینی ترمینال‌ها با متغیرهای جدید، گرامر معادل در فرم نرمال چامسکی را تولید کرد. گرامر معادل در فرم نرمال چامسکی را می‌توان به صورت زیر نوشت:

$$S \to A_1A_2 \mid b$$
$$A_1 \to a$$
$$A_2 \to A_3A_4$$
$$A_3 \to b$$
$$A_4 \to CD$$
$$C \to A_5A_6$$
$$A_5 \to c$$
$$A_6 \to d$$
$$D \to d$$

مثال ۶: گرامر زیر را در نظر بگیرید:

$$S \to AbaB \mid b$$
$$A \to cd$$
$$B \to b$$

ابتدا باید قواعد نامطلوب را از گرامر حذف کرده و سپس با تغییر و جایگزینی ترمینال‌ها با متغیرهای جدید، گرامر معادل در فرم نرمال گریباخ را تولید کرد. گرامر معادل در فرم نرمال گریباخ را می‌توان به این صورت نوشت:

$S \to cA_1A_2A_3B \mid b$

$A \to cA_1$

$A_1 \to d$

$A_2 \to b$

$A_3 \to a$

$B \to b$

ملاحظه می‌شود که برای تبدیل قانون $S \to AbaB$ به فرم گریباخ ابتدا قانون مربوط به A را به فرم گریباخ تبدیل کرده و سپس آن را در S جایگزین می‌کنیم.

نکات و قضایای مهم در رابطه با گرامرهای گریباخ و چامسکی

۱. فرض کنید $G=(V,T,S,P)$ یک گرامر مستقل از متن بدون λ یا قوانین یکه باشد، و نیز فرض کنید k بیشترین تعداد علائم در سمت رأست هر قانون P باشد. آن‌گاه یک گرامر در فرم نرمال چامسکی وجود دارد که حداکثر $|T|+(k-1)|P|$ قانون دارد.

۲. گرامر مستقل از متن $G=(V,T,S,P)$ حاوی قوانین به شکل $A \to x_1Bx_2$ را در نظر بگیرید، فرض کنید که A,B متغیرهای متفاوتی باشند و $B \to y_1 \mid y_2 \mid \ldots y_n$، مجموعه تمامی قوانین P که B را در سمت چپ خود دارند باشد. اگر $\hat{G}=(V,Y,S,\hat{P})$ گرامری باشد که در آن $\hat{P}$ با حذف $A \to x_1Bx_2$ از P با اضافه کردن قوانین زیر به آن بدست آمده باشد، آن‌گاه $L(\hat{G})=L(G)$ است.

$$A \to x_1y_1x_2 \mid x_1y_2x_2 \mid \ldots \mid x_1y_nx_2$$

۳. برای هر گرامر مستقل از متن $G=(V,T,S,P)$ یک گرامر وجود دارد که قوانین آن به صورت زیر باشد:

$$\begin{cases} A \to aBc & A,B,C \in V \\ A \to \lambda & a \in \Sigma \cup \{\lambda\} \end{cases}$$

۴. در یک گرامر نرمال چامسکی، برای تولید یک رشته به طول k به $2k-1$ مرحله اشتقاق نیازمند است.

۵. در یک گرامر گریباخ برای تولید یک رشته به اشتقاقی به اندازه طول رشته نیازمندیم. یعنی اگر طول رشته k باشد، تعداد مراحل اشتقاق برای تولید رشته k مرحله است.

نمونه سؤالات

۱. کدام‌یک از عبارات زیر صحیح است؟

i: حذف قوانین λ ممکن است باعث ایجاد قوانین یکه و یا بی فایده ای شود که قبلاً وجود نداشته است.

ii: حذف قوانین بی فایده ممکن است باعث ایجاد قوانین یکه ای شود که قبلاً وجود نداشته است.

iii: حذف قوانین یکه قطعاً باعث ایجاد قوانین بی فایده ای شود که قبلاً وجود نداشته است.

iv: حذف قوانین یکه ممکن است باعث ایجاد قوانین λ ای شود که قبلاً وجود نداشته است.

۱) i و iv
۲) i و ii و iii
۳) همه موارد
۴) فقط i

۲. کدام‌یک از عبارات زیر نادرست است؟

i: حذف قوانین بی فایده، ممکن است باعث افزایش پیچیدگی گرامر شود.

ii: حذف قوانین λ، همیشه باعث افزایش پیچیدگی گرامر شود.

iii: حذف قوانین یکه، ممکن است باعث شود تا پیچیدگی گرامر ثابت باقی بماند.

iv: حذف قوانین یکه، ممکن است باعث شود تا پیچیدگی گرامر افزایش یابد.

v: حذف قوانین λ، ممکن است باعث شود تا پیچیدگی گرامر ثابت باقی بماند.

vi: حذف قوانین λ، ممکن است باعث افزایش پیچیدگی گرامر شود.

۱) ii و iv و vi
۲) iii و iv و v و vi
۳) i و ii
۴) ii و iii و iv

۳. کدام گزاره صحیح است؟

(علوم کامپیوتر ۸۴)

۱) هر زبان مستقل از متن حتماً می تواند توسط یک گرامر مبهم (ambiguous) تولید شود.

۲) هر زبان مستقل از متن می تواند توسط یک گرامر مستقل از متن تولید شود که برای هر قانون آن مثل $X \to w$ داشته باشیم $|w| \le 2 \left(w \in (V \cup \Sigma)^*\right)$

۳) اگر L یک زبان متناهی باشد حتماً L^* یک زبان نامتناهی است.

۴) تعداد حالات برای اتوماتون های مینیمال زبان های منظم L و L^R همواره برابر است.

۴. کدام زبان مستقل از متن نیست ؟

(علوم کامپیوتر ۸۴)

۱) $\{a^n b a^n b a^n \mid n \geq 0\} \subseteq \{a,b\}^*$

۲) $\{w \mid |w| \equiv 2 (\bmod 5)\} \subseteq \{a,b\}^*$

۳) $\{a^{m+3} b^{2m+1} \mid m \geq 0\} \cup \{a^{3m+1} a^{2m} \mid m \geq 0\} \subseteq \{a,b\}^*$

۴) $\{w \mid w \neq w^R\} \subseteq \{a,b\}^*$

۵. اگر یک گرامر G مستقل از متن و رشته w با طول k متعلق به $L(G)$ باشد. کدام یک از گزینه‌های زیر نادرست است.

۱) اگر گرامر G به شـــکل طبیعی چامســـکی(Chomsky Normal Form) باشد برای اشتقاق (Derivation) w به $2k-1$ مرحله نیاز است.

۲) با حذف قواعدی که به فرم $A \to B$ هستند از گرامر G، ممکن است تعداد مراحل اشتقاق رشته w کمتر شود.

۳) اگر گرامر G به شکل طبیعی گریباخ باشد برای اشتقاق w، حداقل به k مرحله نیاز است.

۴) تعداد مراحل اشتقاق بستگی به فرم گرامر ندارد و قابل پیش بینی نمی باشد.

۶. گرامر $G:(\{S,X,Y\},\{a,b\},S,R)$ را با مجموعه قوانین زیر در نظر بگیرید:

$R:$

$S \to XY$

$S \to a$

$X \to YS \mid b$

$Y \to XS \mid b$

کدام گزاره نادرست است؟

(علوم کامپیوتر ۸۷)

۱) گرامر G یک گرامر مستقل از متن است.

۲) گرامر G به فرم نرمال چامسکی است.

۳) baba توسط گرامر G تولید می‌شود.

۴) baba فقط به یک روش اشتقاق از روی قوانین گرامر G تولید می‌شود.

حل تشریحی

۱. گزینه ۴ درست است.

بر اساس الگوریتم‌های حذف قوانین نامطلوب می‌توان به این نتیجه رسید که گزینه ۴ صحیح است.

۲. گزینه ۳ درست است.

بر اساس الگوریتم‌های حذف قوانین نامطلوب می‌توان به این نتیجه رسید که گزینه ۳ صحیح است.

۳. سؤال دارای گزینه درست نیست.

از آنجا که زبان تهی دارای هیچ رشته‌ای نیست لذا نمی‌توان جمله‌ای از آن پیدا کرد که با یک گرامر مفروض دارای بیش از یک درخت اشتقاق باشد و لذا گزینه ۱ نادرست است. اگر زبان تهی را در نظر نگیریم برای بقیه زبان‌های مستقل از متن گزینه ۱ صحیح است چون می‌توان یک جمله از آن را انتخاب کرده و گرامر را طوری تغییر داد که آن جمله به بیش از یک طریق از روی گرامر قابل اشتقاق باشد. در گزینه ۲ نیز از آنجا که هر زبان مستقل از متن بدون λ می‌تواند توسط یک گرامر در فرم نرمال چامسکی تولید شود لذا برای زبان‌های بدون λ گزینه ۲ درست است ولی در زبان مستقل از متنی که شامل λ باشد گزینه ۲ نادرست است. گزینه ۳ اگر $L = \{\lambda\}$ باشد نادرست است. گزینه ۴ نیز در همه حالات درست نیست و گاهی می‌تواند نادرست باشد.

۴. گزینه ۱ درست است.

در گزینه ۱ نمی‌توان با کمک یک پشته رشته‌های مورد نظر را تشخیص داد و لذا مستقل از متن نیست. زبان گزینه ۲ منظم است و لذا مستقل از متن است. گزینه ۳ نیز از آنجا که اجتماع دو زبان مستقل از متن است پس مستقل از متن است. گزینه ۳ نیز مستقل از متن است و لذا گزینه صحیح ۱ است.

۵. گزینه ۴ درست است.

با توجه به نکات مطرح شده در درس جمله‌های ۱، ۲ و ۳ درست هستند و لذا جمله ۴ نادرست است

۶. گزینه ۴ درست است.

از آنجا که baba دارای یک اشتقاق چپ گرا و یک اشتقاق رأست گرای متفاوت از یکدیگر است لذا گزینه ۴ صحیح است.

خودآزمایی

۱. نشان دهید اگر گرامر G به شکل طبیعی چامسکی(Chomsky Normal Form) باشد برای اشتقاق w (Derivation) به $2k-1$ مرحله نیاز است.

۲. نشان دهید حذف قوانین λ ممکن است باعث ایجاد قوانین یکه و یا بی فایده ای شود که قبلاً وجود نداشته است.

۳. نشان دهید حذف قوانین یکه، ممکن است باعث شود تا پیچیدگی گرامر ثابت باقی بماند.

۴. نشان دهید حذف قوانین یکه، ممکن است باعث شود تا پیچیدگی گرامر افزایش یابد.

۵. نشان دهید حذف قوانین λ ، ممکن است باعث شود تا پیچیدگی گرامر ثابت باقی بماند.

۶. نشان دهید حذف قوانین λ، ممکن است باعث افزایش پیچیدگی گرامر شود.