



تکلیف شماره ۵: پاسخ‌ها

درس نهمی ۸

روش‌های تقدم

۱) گرامر زیر را در نظر بگیرید:

$$S \rightarrow (L) \mid a$$

$$L \rightarrow L, S \mid S$$

برای این گرامر، جدول تجزیه تقدم - عملگر را به دست آورید و با استفاده از آن رشته‌های زیر را تجزیه کنید:

(a, a) $(a, ((a, a), (a, a)))$

پاسخ:

	a	$($	$)$	$,$	$\$$
a			$\cdot >$	$\cdot >$	$\cdot >$
$($	$< \cdot$	$< \cdot$	\doteq	$< \cdot$	
$)$			$\cdot >$	$\cdot >$	$\cdot >$
$,$	$< \cdot$	$< \cdot$	$\cdot >$	$\cdot >$	
$\$$	$< \cdot$	$< \cdot$			

(a, a)

STACK	INPUT	ACTION
$\$$	$(a, a)\$$	shift
$\$ < ($	$a, a)\$$	shift
$\$ < (< a$	$, a)\$$	reduce
$\$ < (N$	$, a)\$$	shift
$\$ < (N < ,$	$a)\$$	shift
$\$ < (N < , < a$	$)\$$	reduce
$\$ < (N < , N$	$)\$$	reduce
$\$ < (N$	$)\$$	shift
$\$ < (N \doteq$	$\$$	reduce
$\$ N$	$\$$	accept

STACK	INPUT	ACTION
\$	(a, (a, a))\$	shift
\$ <	a, (a, a)\$	shift
\$ < (< a	, (a, a)\$	reduce
\$ < (N	, (a, a)\$	shift
\$ < (N < ,	(a, a)\$	shift
\$ < (N < , < (a, a)\$	shift
\$ < (N < , < (< a	, a)\$	reduce
\$ < (N < , < (N	, a)\$	shift
\$ < (N < , < (N < ,	a)\$	shift
\$ < (N < , < (N < , < a)\$	reduce
\$ < (N < , < (N < , N)\$	reduce
\$ < (N < , < (N)\$	shift
\$ < (N < , < (N <))\$	reduce
\$ < (N < , N)\$	reduce
\$ < (N)\$	shift
\$ < (N <)	\$	reduce
\$ N	\$	accept

۲) روابط تقدم عملگر را برای گرامرهای زیر تولید کنید:

$$S \rightarrow aSbS \mid bSaS \mid \epsilon \text{ (الف)}$$

پاسخ:

$$\tilde{G}: S \rightarrow aSbS \mid bSaS \mid \epsilon$$

$$S' \rightarrow \$S\$$$

$$G': S \rightarrow aSbS \mid aSb \mid abS \mid bSaS \mid bSa \mid baS \mid ab \mid ba$$

$$L(G) = L(G') \cup \{\epsilon\}$$

	a	b	\$
a	<	< >	>
b	< >		>
\$	<	<	

$$\text{Leading}(S) = \{a, b\}$$

$$\text{Trailing}(S) = \{a, b\}$$

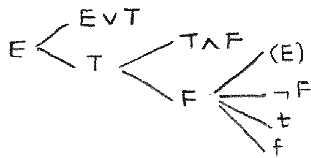
بر دلیل مبهم بودن گرامر، بین بعضی علامت‌چند رابطه تقدم بدست آمده است.

$bexpr \rightarrow bexpr \text{ or } bterm \mid bterm$
 $bterm \rightarrow bterm \text{ and } bfactor \mid bfactor$ (ب)
 $bfactor \rightarrow \text{not } bfactor \mid (bexpr) \mid \text{true} \mid \text{false}$

پاسخ:

ب) $G: E \rightarrow E \vee T \mid T \quad E' \rightarrow \$ E \$$
 $T \rightarrow T \wedge F \mid F$
 $F \rightarrow \neg F \mid (E) \mid t \mid f$

A	Leading(A)	Trailing(A)
E	{ $\vee, \wedge, \neg, (, t, f$ }	{ $\vee, \wedge, \neg,), t, f$ }
T	{ $\wedge, \neg, (, t, f$ }	{ $\wedge, \neg,), t, f$ }
F	{ $\neg, (, t, f$ }	{ $\neg,), t, f$ }



درخت تقدم - شرکت پذیری

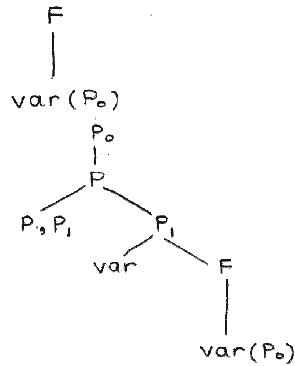
	\vee	\wedge	\neg	()	t	f	\$
\vee	>	<	<	<	>	<	<	>
\wedge	>	>	<	<	>	<	<	>
\neg	>	>	<	<	>	<	<	>
(<	<	<	<	=	<	<	
)	>	>			>			>
t	>	>			>			>
f	>	>			>			>
\$	<	<	<	<		<	<	

۳) یک تجزیه‌گر تقدم ساده برای گرامر زیر بسازید و رشته‌ی $\text{var}(\text{var}, \text{var})$ را با آن تجزیه کنید:

- $F \rightarrow \text{var}(P_0)$
- $P_0 \rightarrow P$
- $P \rightarrow P, P_1 \mid P_1$
- $P_1 \rightarrow \text{var} \mid F$

پاسخ:

- $F' \rightarrow \$F\$$
- $F \rightarrow \text{var}(P_0)$
- $P_0 \rightarrow P$
- $P \rightarrow P, P_1 \mid P_1$
- $P_1 \rightarrow \text{var} \mid F$



A	Head(A)	Tail(A)
F	{var}	{,}
P ₀	{P, P ₁ , F, var}	{P, P ₁ , F, ,}
P	{P, P ₁ , F, var}	{P ₁ , F, var, ,}
P ₁	{F, var}	{F, var, ,}

	F	P ₀	P	P ₁	var	()	,	\$
F						>	>	=	
P ₀						=			
P						>	=		
P ₁						>	>		
var						=	>	>	
(<	=	<	<	<				
)							>	>	>
,	<			=	<				
\$	=				<				

$\text{var}(\text{var}, \text{var})$

STACK	INPUT	ACTION
\$	v(v,v)\$	shift
\$⊗v	(v,v)\$	shift
\$⊗v⊗(v,v)\$	shift
\$⊗v⊗(⊗v	,v)\$	TOP = (, LHS = P ₁ , reduce
\$⊗v⊗(⊗P ₁	,v)\$	TOP = (, LHS = P , reduce
\$⊗v⊗(⊗P	,v)\$	shift
\$⊗v⊗(⊗P⊗,	v)\$	shift
\$⊗v⊗(⊗P⊗,⊗v)\$	TOP = , , LHS = P ₁ , reduce
\$⊗v⊗(⊗P⊗,⊗P ₁)\$	TOP = (, LHS = P , reduce
\$⊗v⊗(⊗P)\$	TOP = (, LHS = P ₀ , reduce
\$⊗v⊗(⊗P ₀)\$	shift
\$⊗v⊗(⊗P ₀ ⊗)	\$	TOP = \$, LHS = F
\$⊗F	\$	accept

۴) یک تجزیه‌گر تقدم ساده برای گرامر زیر بسازید. چگونه باید تداخل‌های جدول را حذف کنیم تا رفتار تجزیه‌گر قاعده‌ی «تطابق هر else با نزدیکترین then تطابق نیافته» را دنبال کند؟

$$\begin{aligned} stmt &\rightarrow \text{if expr then } stmt \\ &| \text{if expr then } stmt \text{ else } stmt \\ &| \text{other} \end{aligned}$$

پاسخ:

 $S \rightarrow \$ stmt \$$
 $stmt \rightarrow \text{if expr then } stmt'$
 $| \text{if expr then } stmt' \text{ else } stmt'$
 $| \text{other}$
 $stmt' \rightarrow stmt$

برای حل مسئله دستگردی، ناپایایی $stmt'$ را وارد می‌کنیم:

 $\text{Head}(stmt) = \{\text{if}, \text{other}\}$
 $\text{Tail}(stmt) = \{stmt', \text{other}\}$
 $\text{Head}(stmt') = \{stmt, \text{if}, \text{other}\}$
 $\text{Tail}(stmt') = \{stmt, stmt', \text{other}\}$

	if	then	else	other	expr	stmt	stmt'	\$
if					⊖			
then	⊖			⊖		⊖	⊖	
else	⊖			⊖		⊖	⊖	
other			⊗					⊗
expr		⊖						
stmt			⊖, ⊗					⊖
stmt'			⊗					⊗
\$	⊖			⊖		⊖		

تداخل بین $stmt$ و $else$ پیش آمده است. رفتار مطلوب، تطبیق نزدیک‌ترین $else$ تطبیق نیافته با یک $then$ است. پس ⊖ را انتخاب می‌کنیم.

۵) الگوریتم‌های مناسبی برای محاسبه‌ی توابع زیر ارایه دهید که در آنها A یک ناپایانه است.

$$\bullet \text{FirstTerm}(A)$$

$$\bullet \text{LastTerm}(A)$$

$$\bullet \text{Head}(A)$$

$$\bullet \text{Tail}(A)$$

پاسخ: الگوریتم‌های محاسبه‌ی $\text{FirstTerm}(A)$ و $\text{Head}(A)$ بسیار شبیه الگوریتم محاسبه‌ی $\text{First}(A)$ است. با معکوس کردن سمت راست قواعد گرامر، می‌توان از $\text{FirstTerm}(A)$ و $\text{Head}(A)$ برای محاسبه‌ی $\text{LastTerm}(A)$ و $\text{Tail}(A)$ استفاده کرد.

۶) نشان دهید هر گرامر مستقل از متن می‌تواند به یک گرامر عملگر (operator-grammar) تبدیل شود که قواعد تولید آن به یکی از صورت‌های زیر است:

$$A \rightarrow aBcC, A \rightarrow aBb, A \rightarrow aB, A \rightarrow a$$

و اگر ϵ در زبان وجود داشته باشد، $S \rightarrow \epsilon$ هم یک قاعده‌ی تولید باشد.

پاسخ: ابتدا گرامر مستقل از متن داده شده را به فرم نرمال گریباخ تبدیل می‌کنیم (رک). به درس نظریه‌ی زبان‌ها و ماشین‌ها: درس‌نامه ۶). در این صورت تبدیل گرامر به گرامری با قواعد به فرم خواسته شده، سراسر است می‌باشد.

۷) آیا گرامر زیر، یک گرامر عملگر است؟ چرا؟ در صورت منفی بودن پاسخ، معادل عملگر آن را بنویسید:

$$E \rightarrow EAE \mid (E) \mid -E \mid \text{id}$$

$$A \rightarrow + \mid - \mid * \mid /$$

پاسخ: خیر، اما گرامر زیر معادل با گرامر بالاست و یک گرامر عملگر است:

$$E \rightarrow E + E \mid E - E \mid E * E \mid E / E \mid (E) \mid \text{id}$$

در به دست آوردن این گرامر جدید، از قاعده‌ی جایگزینی استفاده کرده‌ایم. ملاحظه می‌کنیم که در هیچ قاعده‌ای دو ناپایانه‌ی مجاور دیده نمی‌شود.