طراحی کامپایلرها

نیم‌سال دوم ۹۹ـ۹۸

دانشکده‌ی مهندسی کامپیوتر

نام و نام خانوادگی: حسن ذاکر، علیرضا دقیق، سپهر فعلی

پاسخ تمرین سری دوم | پارسرها | موعد تحویل: ۹۸/۱۲/۲۵

ما ۳ نفر همه سوالات را با هم حل کردیم

مسئله‌ی 1. گرامر مبهم

پاسخ.

- $S \rightarrow As \mid \epsilon$
  $A \rightarrow 01B \mid 0A1$
  $B \rightarrow 1B \mid \epsilon$

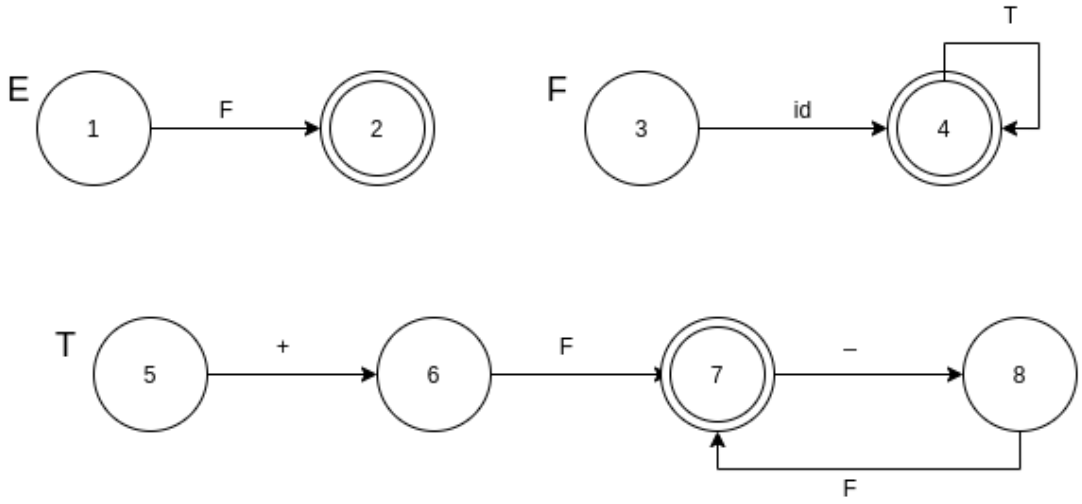
- ابهام ویژگی یک گرامر است نه زبان. ابهام در یک گرامر این قابلیت را ایجاد می‌کند که برای یک ورودی چند درخت پارس بتوان تشکیل داد و ممکن است بر حسب قواعد معنایی جوابی خارج که مورد نظرمان نیست را با پارس کردن درختی دیگ ایجاد کند. گاهی هم ممکن است نتوان پارس را انجام داد.

مسئله‌ی 2. عبارت آرمانی

پاسخ.

۱. علاوه بر پارس‌استک PS یک استک جدید به نام TS تعریف می‌کنیم که در هنگام پارس کردن به کمک ما می‌آید. به این صورت که هر موقع + داشتیم آن را به TS اضافه می‌کنیم و هرگاه به ـ رسیدیم از استک POP TS می‌کنیم. عملیات پارس به‌صورت معمول با PS انجام می‌شود و علاوه بر حالت‌های invalid که ممکن است در PS رخ دهد زمانی که باید از POP TS کنیم و TS خالی باشد نیز باید حالت invalid تشخیص داده‌شود.

۲. از آنجایی که بعد از دیدن ـ حتما یک‌بار باید F را ببینیم و اگر id دیگری بخواهد بیاید حتما باید T را ببینیم و در ابتدای T حتما + می‌بینیم، حالت‌های valid پارس خواهند شد و حالت‌های invalid قابل پارس شدن با استفاده از PS نیستند.



مسئله‌ی 3. **Recursive Descent**

پاسخ. چون این گرامر left recursive است قابلیت پارس با Recursive descent را ندارد. برای جل این مشکل ابتدا آن‌را رفع ابهام می‌کنیم سپس شبه‌کد مربوط به آن‌را می‌زنیم.

$E \rightarrow T + E \mid T - E \mid T$
$T \rightarrow 11T' \mid 10T'$
$T' \rightarrow 10T \mid 11T \mid \epsilon$

```
int main()
{
    // E is a start symbol.
    E();

    // if lookahead = $, it represents the end of the string
    // Here l is lookahead.
    if (l == '$')
        printf("Parsing Successful");
}

// Definition of E, as per the given production
E()
{
    T();
    if (l == '+') {
        match('+');
        E();
    } else if (l == '-'){
        match('-');
        E();
    }
```

```
    return();
}

// Definition of E' as per the given production
T()
{
    if (l == '1') {
        match('1');
        if(l == '1'){
            match('1');
            T'();
        }else if(l == '0'){
            match('0');
            T'();
        }
    }

}

T'()
{
    if (l == '1') {
        match('1');
        if(l == '1'){
            match('1');
            T();
        }else if(l == '0'){
            match('0');
            T();
        }
    }else{
        return();
    }
}

// Match function
match(char t)
{
    if (l == t) {
        l = getchar();
    }
    else
        printf("Error");
}
```

مسئله‌ی 4. پرانتزگذاری معتبر

پاسخ. بله $\mathrm{LL}(1)$ است.

Follow	First	
) $	ϵ (	S
($	(	P

$	)	(	
$\epsilon \to$ S	$\epsilon \to$ S	PS $\to$ S	S
		(S) $\to$ P	P

مسئله‌ی 5. با طعم اپسیلون

پاسخ.

۱.

A $\to$ aB
B $\to$ bG
G $\to$ cH | D | ϵ
H $\to$ C | d
C $\to$ c | ϵ
D $\to$ dE
E $\to$ cC | ϵ

۲.

Follow	First	
$	a	A
$	b	B
$	ϵ d c	G
$	ϵ d c	H
$	ϵ c	C
$	d	D
$	ϵ c	E

۳.

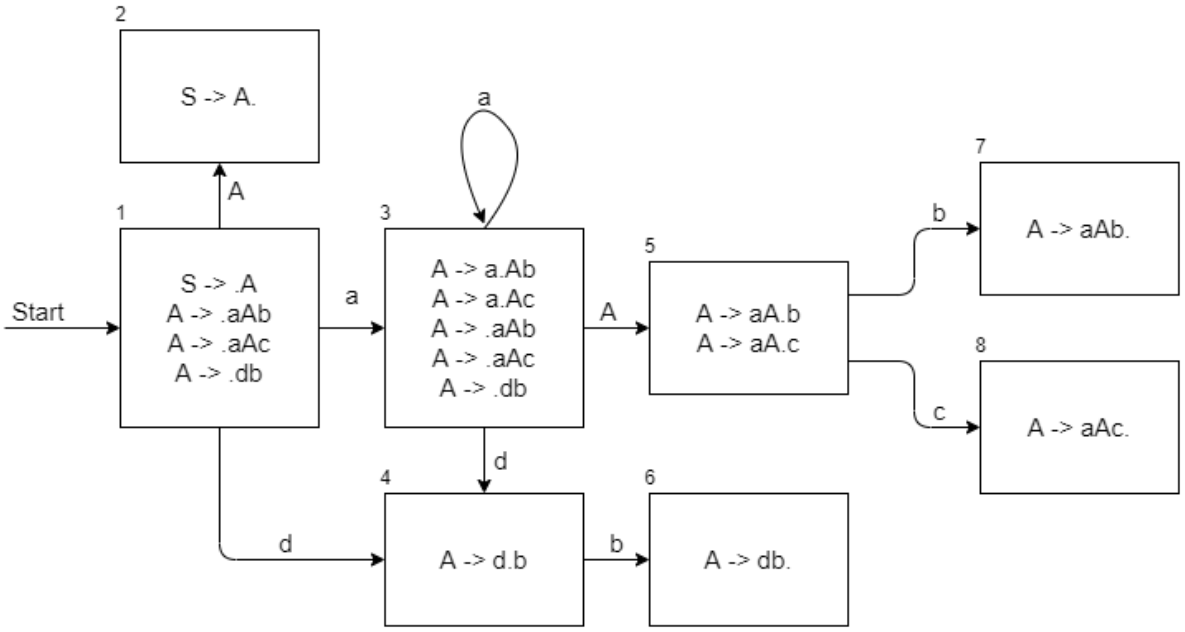
$	d	c	b	a	
				1	A
			2		B
5	4	3			G
6	7	6			H
9		8			C
	10				D
12		11			E

.۴

abdcc$	A$
abdcc$	aB$
bdcc$	B$
bdcc$	bG$
dcc$	G$
dcc$	D$
dcc$	dE$
cc$	E$
cc$	cC$
c$	C$
c$	c$
$	$

مسئله‌ی 6. پارسر LR(۰)

پاسخ.



۱.

A	d	c	b	a	
G2	S4			S3	1
acc	acc	acc	acc	acc	2
G5	S4			S3	3
			S6		4
		S8	S7		5
R4	R4	R4	R4	R4	6
R2	R2	R2	R2	R2	7
R3	R3	R3	R3	R3	8

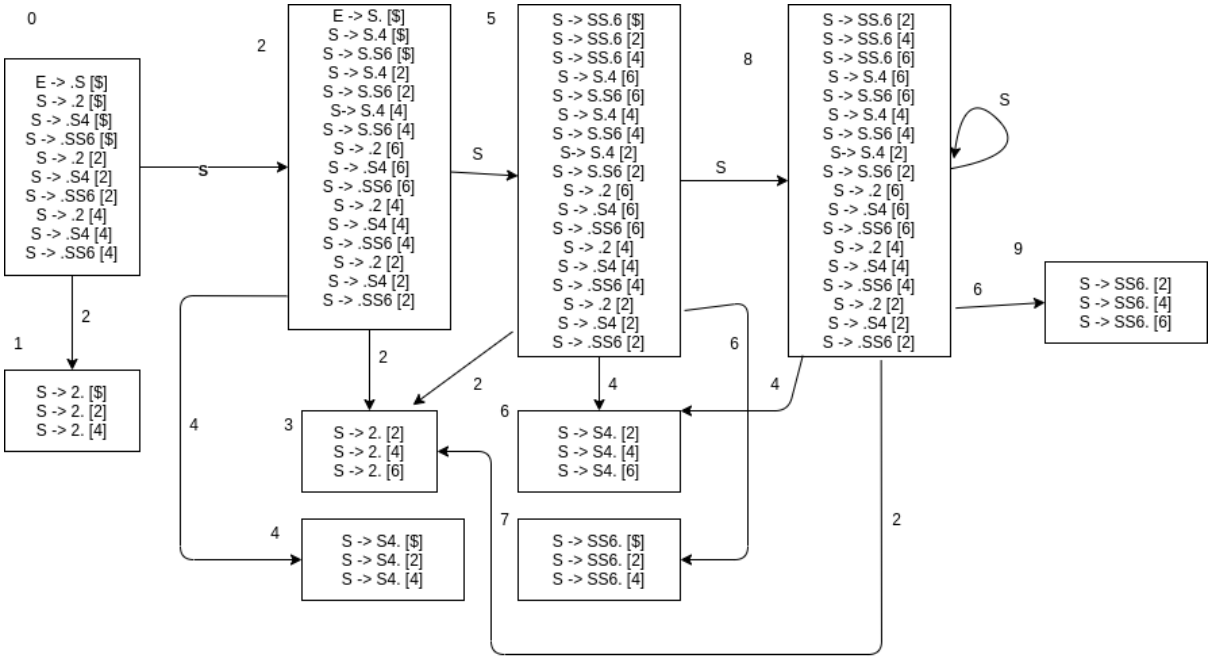
۲. ابتدا در حالت ۱ قرار داریم.(a|adbc)
با خواندن a به حالت ۳ میرویم و توکن بعدی را میخوانیم. (aa|dbc)
در این حالت نیز، باز با خواندن a به حالت ۳ میرویم و توکن بعدی را میخوانیم. (aad|bc)
حال با خواندن d به حالت ۴ میرویم و توکن بعدی را میخوانیم. (aadb|c)
حال با خواندن b به حالت ۶ میرویم که در این حالت، reduction داریم به صورت $A \rightarrow db$ پس ۲ حالت به عقب میرویم(حالت ۳) و db را نیز به A تبدیل میکنیم.(aaA|c)
حال در حالت ۳ هستیم و با خواندن A به حالت ۵ میرویم. توکن بعدی را میخوانیم. (aaAc|)
حال با خواندن c به حالت ۸ میرویم که در این حالت، reduction داریم به صورت $A \rightarrow aAc$ پس ۳ حالت به عقب میرویم(حالت ۳) و aAc را نیز به A تبدیل میکنیم. (aA|)
حال در حالت ۳ هستیم و با خواندن A به حالت ۵ میرویم.و همین جا گیر میفتیم.چرا که aA را داریم و دیگر هیچ.
پس نمیتوان این رشته را پارس کرد.

مسئله‌ی 7. LALR یا !SLR

پاسخ. اثبات به برهان خلف. فرض خلف : slr(۱) بدون conflict داریم اما lalr(۱) بدون conflict نداریم. از آنجایی که در lalr۱ برخورد داشته ایم پس یک terminal وجود دارد که در آنجا هم می‌توانیم کاهش دهیم هم شیفت دهیم. پس یک قاعده تولید وجود دارد که terminal ذکر شده در بالا در سمت راست فرضیnonterminal وجود دارد پس این nonterminal در فالو خود terminal ذکر شده را دارد. پس در slr۱ هم .داریمconflict تناقض.

مسئله‌ی 8. پارسر LR(۱)

پاسخ.



S	&	۶	۴	۲	
G۲				S۱	۰
	R۱		R۱	R۱	۱
G۵	acc		S۴	S۳	۲
		R۱	R۱	R۱	۳
	R۲		R۲	R۲	۴
G۸		S۷	S۶	S۳	۵
		R۲	R۲	R۲	۶
	R۳		R۳	R۳	۷
G۸		S۹	S۶	S۳	۸
		R۳	R۳	R۳	۹

ابتدا در state شماره ۰ هستیم. ۲ را می‌بینیم و بهstate شماره ۱ می‌رویم. سپس دوی بعدی را به عنوان lookahead انتخاب می‌کنیم. در s۱ به ازای head a look ۲ عمل reduce با قاعده تولید شماره یک را داریم.پس به ۲ و s۱ را خط می‌زنیم و S را جایگزین می‌کنیم. همین روند را تا آخر به صورت زیر ادامه می‌دهیم

۰ -> ۲ -> S۱ ، head a look = ۲
۰ -> S -> G۲ -> ۲ -> S۳ ، head a look = ۶
۰ -> S -> G۲ -> S -> G۵ -> ۶ -> S۷ ، head a look = ۴
۰ -> S -> G۲ -> ۴ -> S۴ ، ahead look = ۲
۰ -> S -> G۲ -> ۲ -> S۳ ، head a look = ۲
۰ -> S -> G۲ -> S -> G۵ -> ۲ -> S۳ ، head a look = ۴
۰ -> S -> G۲ -> S -> G۵ -> S -> G۸ -> ۴ -> S۶ ، head a look = ۶
۰ -> S -> G۲ -> S -> G۵ -> S -> G۸ -> ۶ -> S۹ < _ ، head a look = ۶
۰ -> S -> G۲ -> S -> G۵ -> ۶ -> S۷ ، head a look = &
۰ -> S G۲ -> -> & -> Accept

مسئله‌ی 9. پارسر LALR(۱)

پاسخ.

مسئله‌ی 10. مقایسه LR(۱) و SLR(۱)

پاسخ.