

Noncanonical LALR(1) Parsing

Sylvain Schmitz
schmitz@i3s.unice.fr



June 26, 2006

JAVA Declarations

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$$\begin{aligned} \langle FDecl \rangle &\rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \\ \langle FModS \rangle &\rightarrow \langle FModS \rangle \langle FMod \rangle \mid \varepsilon \\ \langle FMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{transient} \mid \dots \\ \langle Type \rangle &\rightarrow \langle NonVoidType \rangle \\ \langle MHead \rangle &\rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle \\ \langle MModS \rangle &\rightarrow \langle MModS \rangle \langle MMod \rangle \mid \varepsilon \\ \langle MMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{abstract} \mid \dots \\ \langle ResType \rangle &\rightarrow \langle NonVoidType \rangle \mid \text{void} \end{aligned}$$

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$$\begin{aligned} \langle FDecl \rangle &\rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \\ \langle FModS \rangle &\rightarrow \langle FModS \rangle \langle FMod \rangle \mid \varepsilon \\ \langle FMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{transient} \mid \dots \\ \langle Type \rangle &\rightarrow \langle NonVoidType \rangle \\ \langle MHead \rangle &\rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle \\ \langle MModS \rangle &\rightarrow \langle MModS \rangle \langle MMod \rangle \mid \varepsilon \\ \langle MMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{abstract} \mid \dots \\ \langle ResType \rangle &\rightarrow \langle NonVoidType \rangle \mid \text{void} \end{aligned}$$

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$\langle FDecl \rangle \rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ;$

$\langle FModS \rangle \rightarrow \langle FModS \rangle \langle FMod \rangle | \varepsilon$

$\langle FMod \rangle \rightarrow \text{public} | \text{final} | \text{static} | \text{transient} | \dots$

$\langle Type \rangle \rightarrow \langle NonVoidType \rangle$

$\langle MHead \rangle \rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle$

$\langle MModS \rangle \rightarrow \langle MModS \rangle \langle MMod \rangle | \varepsilon$

$\langle MMod \rangle \rightarrow \text{public} | \text{final} | \text{static} | \text{abstract} | \dots$

$\langle ResType \rangle \rightarrow \langle NonVoidType \rangle | \text{void}$

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$$\begin{aligned} \langle FDecl \rangle &\rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \\ \langle FModS \rangle &\rightarrow \langle FModS \rangle \langle FMod \rangle \mid \varepsilon \\ \langle FMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{transient} \mid \dots \\ \langle Type \rangle &\rightarrow \langle NonVoidType \rangle \\ \langle MHead \rangle &\rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle \\ \langle MModS \rangle &\rightarrow \langle MModS \rangle \langle MMod \rangle \mid \varepsilon \\ \langle MMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{abstract} \mid \dots \\ \langle ResType \rangle &\rightarrow \langle NonVoidType \rangle \mid \text{void} \end{aligned}$$

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$$\begin{aligned} \langle FDecl \rangle &\rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \\ \langle FModS \rangle &\rightarrow \langle FModS \rangle \langle FMod \rangle \mid \varepsilon \\ \langle FMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{transient} \mid \dots \\ \langle Type \rangle &\rightarrow \langle NonVoidType \rangle \\ \langle MHead \rangle &\rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle \\ \langle MModS \rangle &\rightarrow \langle MModS \rangle \langle MMod \rangle \mid \varepsilon \\ \langle MMod \rangle &\rightarrow \text{public} \mid \text{final} \mid \text{static} \mid \text{abstract} \mid \dots \\ \langle ResType \rangle &\rightarrow \langle NonVoidType \rangle \mid \text{void} \end{aligned}$$

JAVA Grammar

Gosling et al. [1996]

Example

```
public class Declarations {
    public final static int FIELD = 10;
    public static int method (int i) {
        // ...
    }
}
```

$\langle FDecl \rangle \rightarrow \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ;$

$\langle FModS \rangle \rightarrow \langle FModS \rangle \langle FMod \rangle | \varepsilon$

$\langle FMod \rangle \rightarrow \text{public} | \text{final} | \text{static} | \text{transient} | \dots$

$\langle Type \rangle \rightarrow \langle NonVoidType \rangle$

$\langle MHead \rangle \rightarrow \langle MModS \rangle \langle ResType \rangle \langle MDecl \rangle \langle Throws \rangle$

$\langle MModS \rangle \rightarrow \langle MModS \rangle \langle MMod \rangle | \varepsilon$

$\langle MMod \rangle \rightarrow \text{public} | \text{final} | \text{static} | \text{abstract} | \dots$

$\langle ResType \rangle \rightarrow \langle NonVoidType \rangle | \text{void}$

Deterministic Parsing

+ Performance

+ Safety

— Generality

LR(k) Parsing

Knuth [1965]

- ▶ complete stack information
- ▶ k symbols of lookahead

LR(k) Parsing

Knuth [1965]

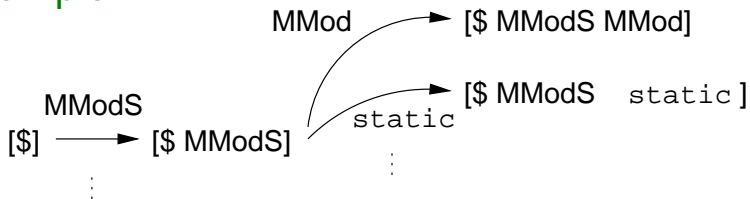
- ▶ complete stack information: **regular language**
- ▶ k symbols of lookahead

LR(k) Parsing

Knuth [1965]

- ▶ complete stack information: **regular language**
- ▶ k symbols of lookahead

Example



Nondeterminism

Example

```
public final static int FIELD = 10 ;
```

```
[$] || public final static int FIELD = 10;$
```

Nondeterminism

Example

MModS

ϵ public final static int FIELD = 10 ;

$[\$]\epsilon$ || public final static int FIELD = 10;\$

$\stackrel{\text{reduce}}{=} [\$][\$ <MModS>]$ || public final static int FIELD = 10;\$

Nondeterminism

Example

MModS

ε public final static int FIELD = 10 ;

$[\$][\$ \langle MModS \rangle] \parallel \text{public final static int FIELD} = 10; \$$

$\stackrel{\text{shift}}{\parallel} [\$][\$ \langle MModS \rangle] [\$ \langle MModS \rangle \text{public}] \parallel \text{final static int FI}$

Nondeterminism

Example

$MModS$ $MMod$
 | |
 ϵ $public$ $final$ $static$ int $FIELD = 10 ;$

$[\$][\$ <MModS>][\$ <MModS> public] \parallel final static int F$

$\stackrel{\text{reduce}}{=} [\$][\$ <MModS>][\$ <MModS> <MMod>] \parallel final static int$

Nondeterminism

Example



$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle MMod \rangle] \parallel$ final static int

$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle] \parallel$ final static int FIELD = 10;\$

Nondeterminism

Example



$[\$][\$ \langle MModS \rangle] \parallel \text{final static int FIELD} = 10; \$$

$\stackrel{\text{shift}}{\parallel} [\$][\$ \langle MModS \rangle] [\$ \langle MModS \rangle \text{final}] \parallel \text{static int FIELD} =$

Nondeterminism

Example

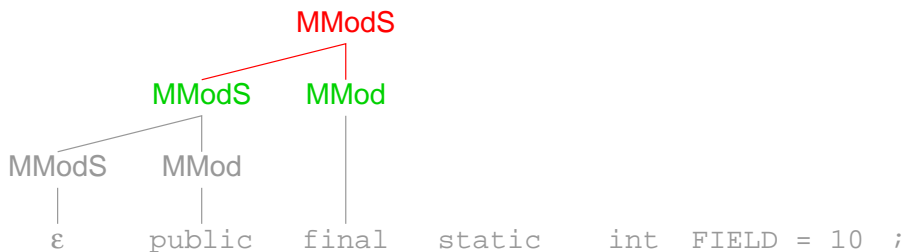


$$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \text{final}] \parallel \text{static int FIELD} =$$

$$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle MMod \rangle] \parallel \text{static int FIELD}$$

Nondeterminism

Example

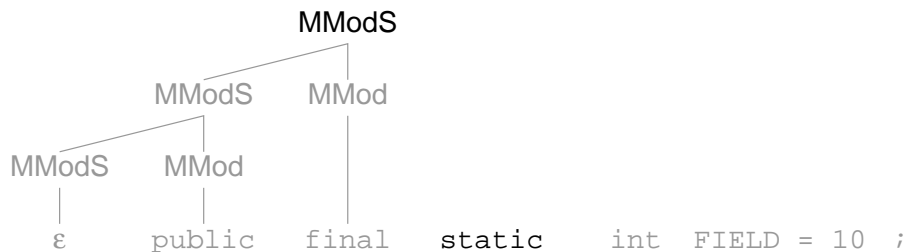


$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle MMod \rangle] \parallel \text{static int FIELD}$

$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle] \parallel \text{static int FIELD} = 10; \$$

Nondeterminism

Example

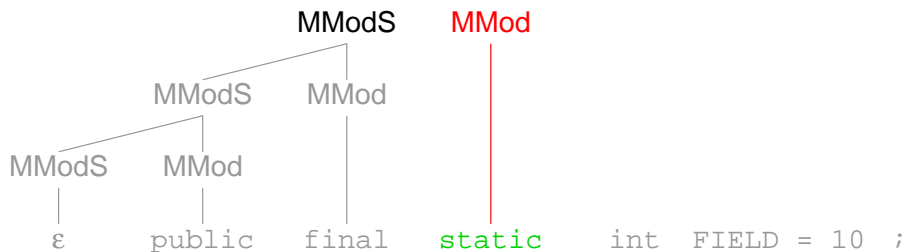


`[$][$ <MModS>] || static int FIELD = 10;$`

$\stackrel{\text{shift}}{=} [\$][\$ <MModS>] [\$ <MModS> \text{static}] || \text{int FIELD} = 10; \$$

Nondeterminism

Example

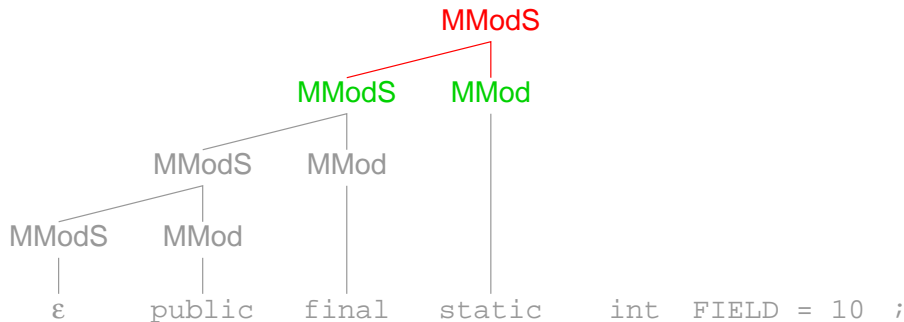


$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \text{static}] \parallel \text{int FIELD} = 10; \$$

$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle MMod \rangle] \parallel \text{int FIELD} = 10; \$$

Nondeterminism

Example

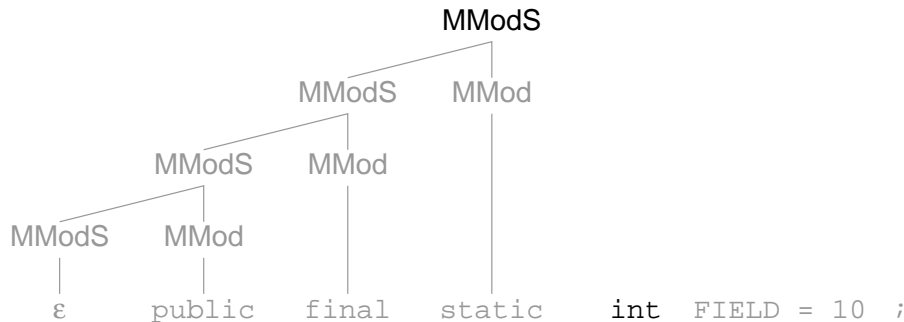


$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle MMod \rangle] \parallel \text{int FIELD} = 10;$

$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle] \parallel \text{int FIELD} = 10; \$$

Nondeterminism

Example

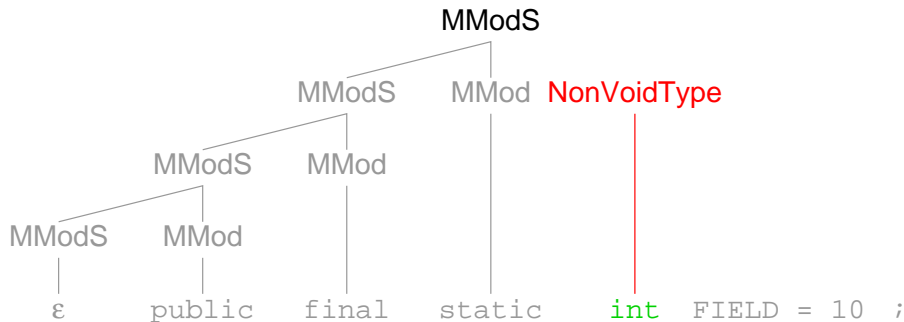


`[$][$ <MModS>] || int FIELD = 10;$`

shift `[$][$ <MModS>] [$ <MModS> int] || FIELD = 10;$`

Nondeterminism

Example

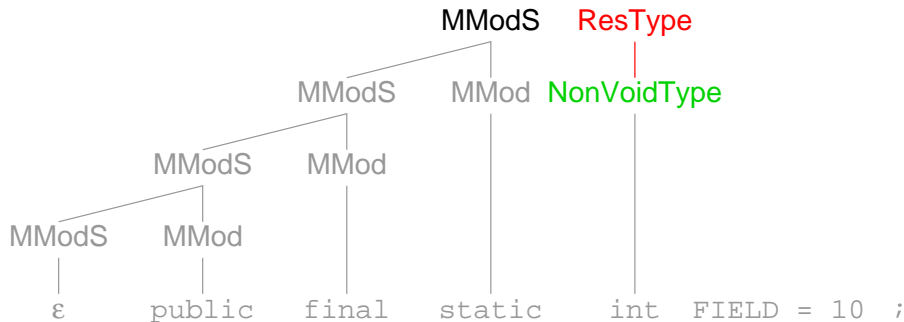


$$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \text{int}] \parallel \text{FIELD} = 10; \$$$

$$\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle NonVoidType \rangle] \parallel \text{FIELD} = 10; \$$$

Nondeterminism

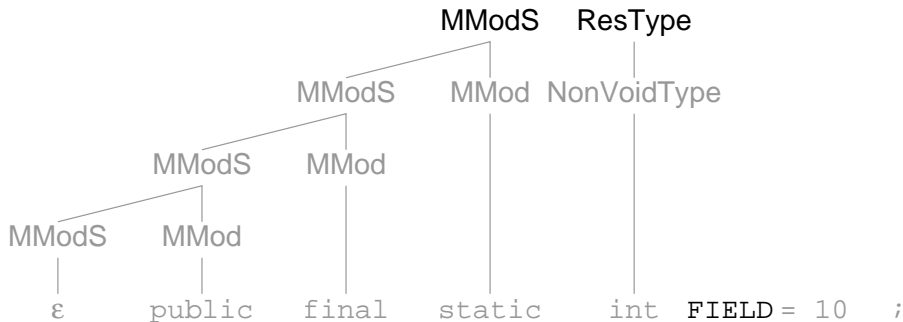
Example



$[\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle NonVoidType \rangle] \parallel \text{FIELD} = 1$
 $\stackrel{\text{reduce}}{=} [\$][\$ \langle MModS \rangle][\$ \langle MModS \rangle \langle ResType \rangle] \parallel \text{FIELD} = 10; \$$

Nondeterminism

Example

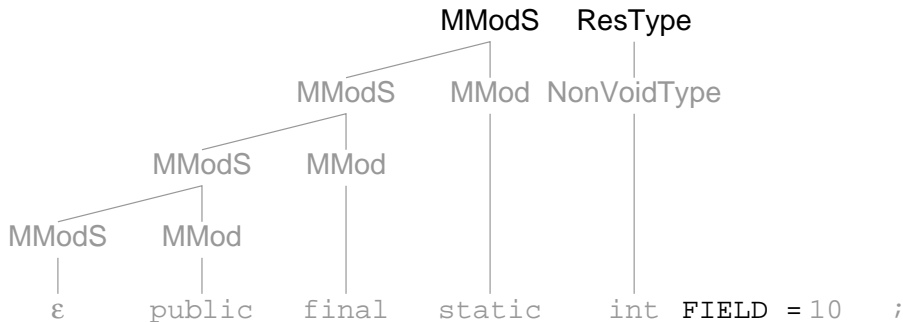


$[\$] \dots [\$ \langle MModS \rangle \langle ResType \rangle] \parallel \text{FIELD} = 10; \$$

$\underset{\text{shift}}{\parallel} [\$] \dots [\$ \langle MModS \rangle \langle ResType \rangle \text{FIELD}] \parallel = 10; \$$

Nondeterminism

Example

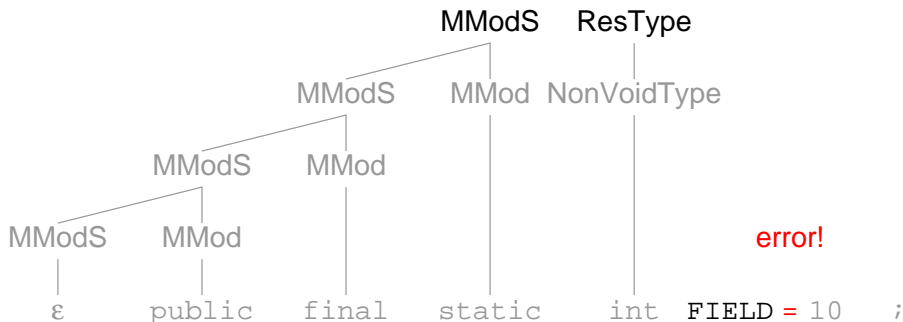


$[\$] \dots [\$ \langle MModS \rangle \langle ResType \rangle FIELD] \parallel = 10; \$$

$\underset{\text{shift}}{\parallel} [\$] \dots [\$ \langle MModS \rangle \langle ResType \rangle FIELD =] \parallel 10; \$$

Nondeterminism

Example



$[\$] \dots [\langle MModS \rangle \langle ResType \rangle \text{FIELD} =] \parallel 10; \$$

Nondeterminism

Example

FModS

ε public final static int FIELD = 10 ;

$[\$]\epsilon \parallel$ public final static int FIELD = 10;\$

$\stackrel{\text{reduce}}{=} [\$][\$ \langle \text{FModS} \rangle] \parallel$ public final static int FIELD = 10;\$

LR(k) Parsing

- ▶ complete stack information
- ▶ k symbols of lookahead

Example

Choice between reductions $\langle FModS \rangle \rightarrow \epsilon$ and $\langle MModS \rangle \rightarrow \epsilon$:

```
[$]ε||public static final ... final static int l =  
[$]ε||public static final ... final static int l (
```

- ▶ Unbounded lookahead needed

LR(k) Parsing

- ▶ complete stack information
- ▶ k symbols of lookahead

Example

Choice between reductions $\langle FModS \rangle \rightarrow \varepsilon$ and $\langle MModS \rangle \rightarrow \varepsilon$:

```
[$]ε||public static final ... final static int l =  
[$]ε||public static final ... final static int l (
```

- ▶ Unbounded lookahead needed

LR(k) Parsing

- ▶ complete stack information
- ▶ k symbols of lookahead

Example

Choice between reductions $\langle FModS \rangle \rightarrow \varepsilon$ and $\langle MModS \rangle \rightarrow \varepsilon$:

```
[$]ε||public static final ... final static int l =  
[$]ε||public static final ... final static int l (
```

- ▶ Unbounded lookahead needed

Parsing a Program Source

- ▶ Determinism: safety
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity

Parsing a Program Source

- ▶ Determinism: safety
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity

Noncanonical parsing

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

```
public final static int FIELD = 10 ;
```

```
$ || public final static int FIELD = 10;$
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

```
public    final    static    int    FIELD = 10 ;
```

```
$ || public final static int FIELD = 10;$
```

```
   $ public || final static int FIELD = 10;$  
shift
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

```
public    final    static    int    FIELD = 10 ;
```

```
$public || final static int FIELD = 10;$
```

```
shift $public final || static int FIELD = 10;$
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

```
public    final    static    int    FIELD = 10 ;
```

```
$public final || static int FIELD = 10;$
```

```
⏟  
|shift $public final static || int FIELD = 10;$
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

```
public final static int FIELD = 10 ;
```

```
$public final static || int FIELD = 10;$
```

```
⏟  
|shift $public final static int || FIELD = 10;$
```


Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

NonVoidType

public final static int FIELD = 10 ;

\$public final static int || FIELD = 10;\$

$\stackrel{\text{reduce}}{=}$ \$public final static || *<NonVoidType>* FIELD = 10;\$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

NonVoidType

```
public final static int FIELD = 10 ;
```

```
$public final static || <NonVoidType> FIELD = 10;$
```

```
shift $public final static <NonVoidType> || FIELD = 10;$
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

NonVoidType

```
public final static int FIELD = 10 ;
```

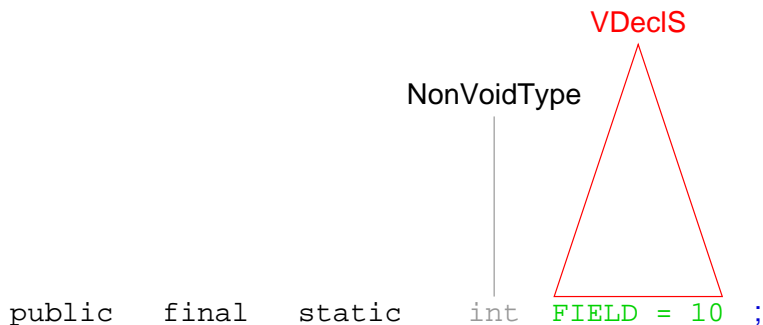
```
$public final static <NonVoidType> || FIELD = 10 ;$
```

```
shift $public final static <NonVoidType> FIELD = 10 || ;$
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



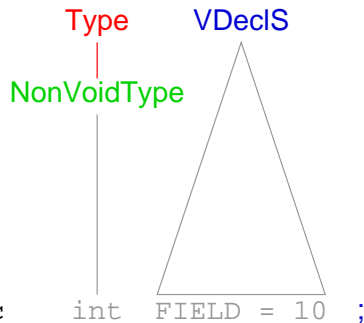
`$public final static <NonVoidType> FIELD = 10 || ;$`

$\stackrel{\text{reduce}}{=} \text{\$public final static <NonVoidType> || <VDecls> ;\$}$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



```
public final static int FIELD = 10 ;
```

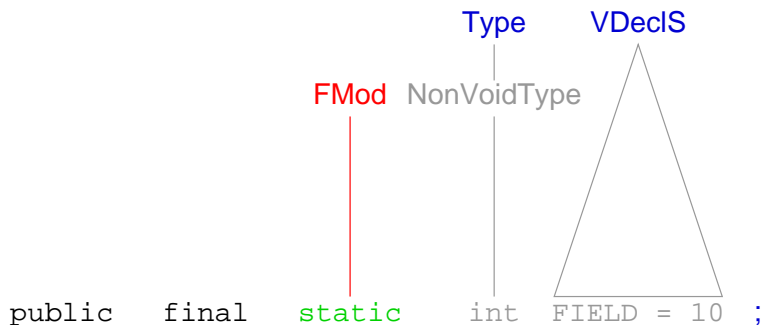
```
$public final static <NonVoidType> || <VDecls> ;$
```

```
 $\stackrel{\text{reduce}}{=} \$public final static || <Type> <VDecls> ;$$ 
```

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



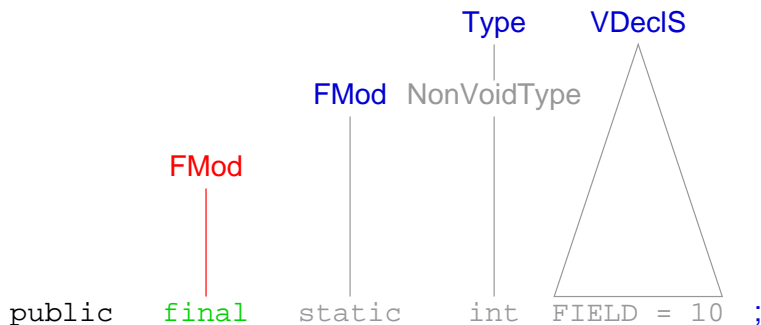
`$public final static || <Type><VDecls> ;$`

$\stackrel{\text{reduce}}{=} \$public final || <FMod> <Type><VDecls> ;$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



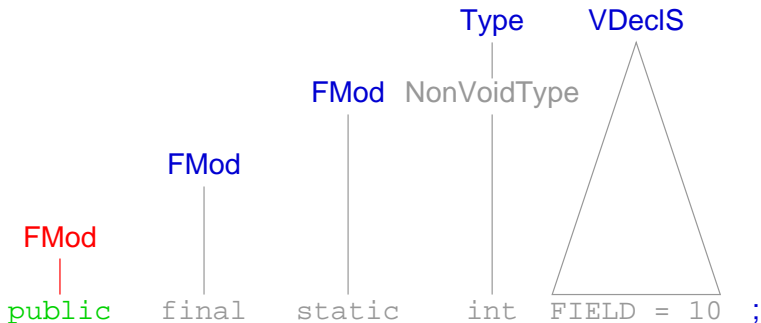
`$public final || <FMod><Type><VDecls> ;$`

$\stackrel{\text{reduce}}{=} \text{\$public || } <FMod> <FMod><Type><VDecls> ;\text{\$}$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



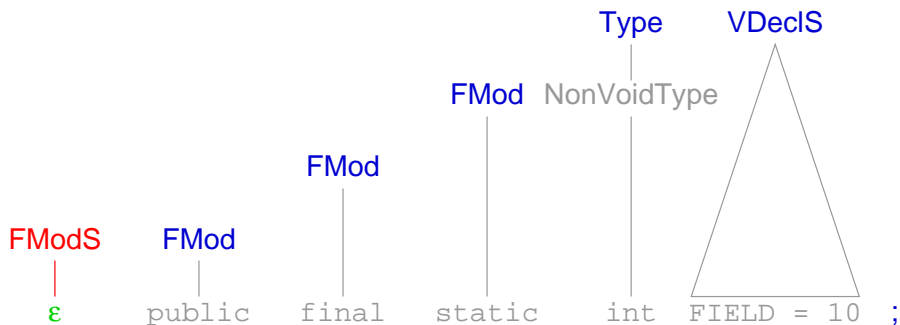
$\$$ `public` \parallel $\langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

$\stackrel{\text{reduce}}{=} \$ \parallel \langle FMod \rangle \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



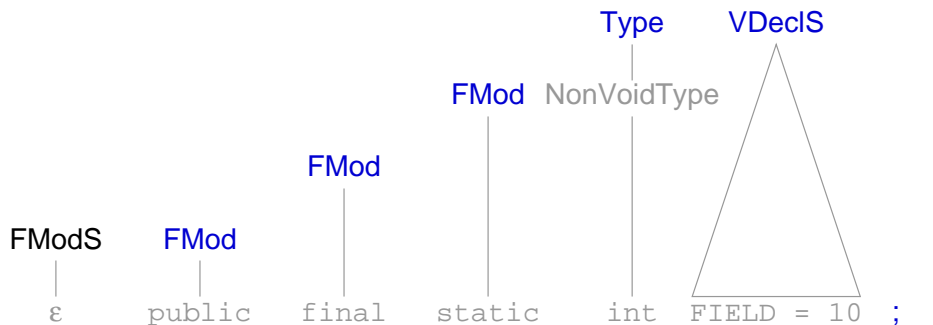
$\$ \epsilon \parallel \langle FMod \rangle \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle ; \$$

$\stackrel{\text{reduce}}{=} \$ \parallel \langle FModS \rangle \langle FMod \rangle \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



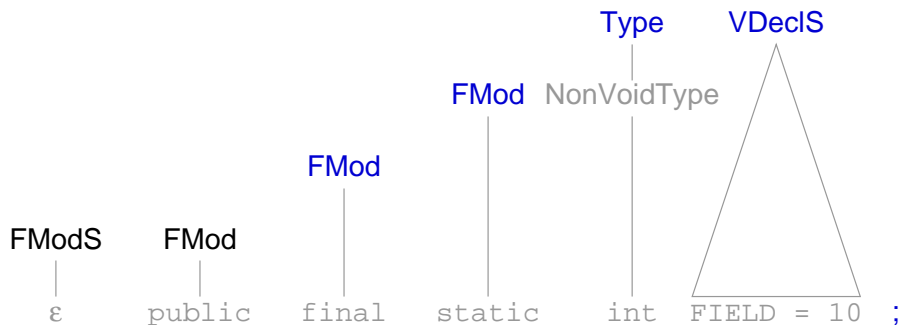
\$ || *<FMModS>* *<FMMod>**<FMMod>**<FMMod>**<Type>**<VDecls>*

shift \$ *<FMModS>* || *<FMMod>**<FMMod>**<FMMod>**<Type>**<VDecls>*

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

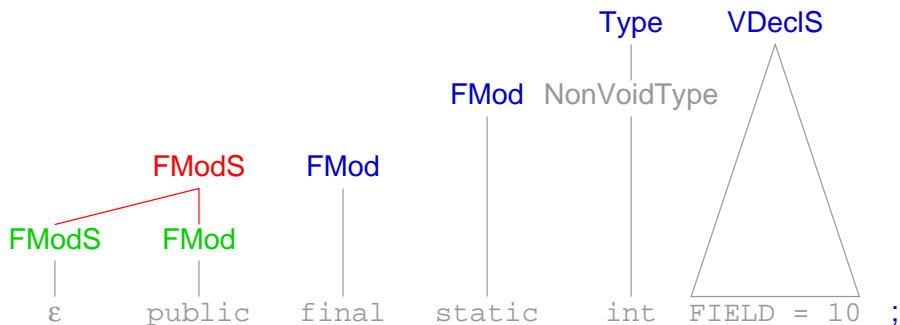


$\$ \langle FModS \rangle \parallel \langle FMod \rangle \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle$
 $\text{shift } \$ \langle FModS \rangle \langle FMod \rangle \parallel \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

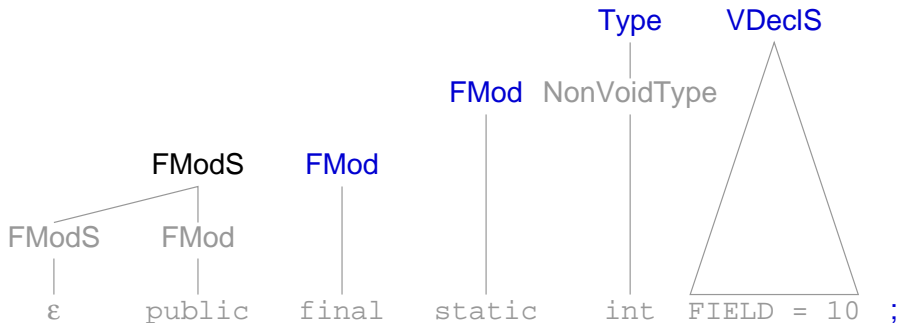


$\$ \langle FModS \rangle \langle FMod \rangle \parallel \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle$
 $\stackrel{\text{reduce}}{=} \$ \parallel \langle FModS \rangle \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



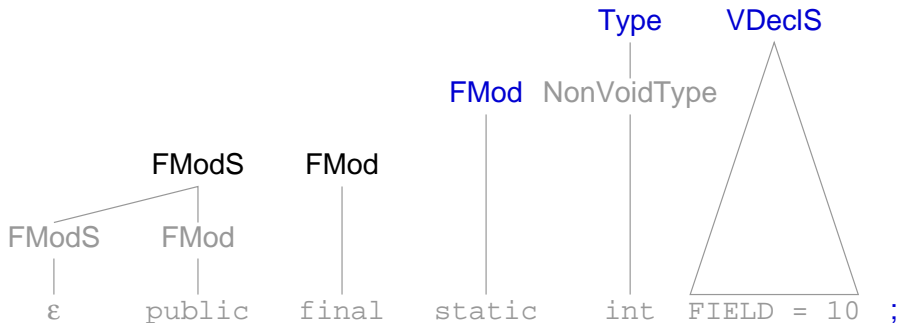
$\$ \parallel \langle FModS \rangle \quad \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle ; \$$

$\underset{\text{shift}}{\parallel} \$ \langle FModS \rangle \parallel \langle FMod \rangle \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

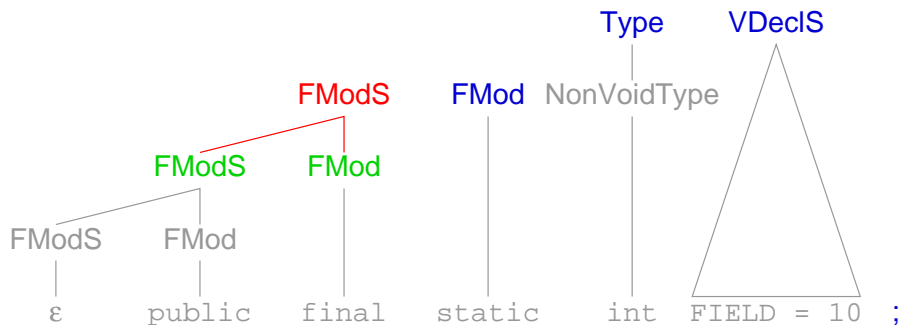


$$\begin{array}{l}
 \$ \langle FModS \rangle \quad || \quad \langle FMod \rangle \quad \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle \quad ; \$ \\
 \text{shift} \quad \$ \langle FModS \rangle \quad \langle FMod \rangle \quad || \quad \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle \quad ; \$
 \end{array}$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



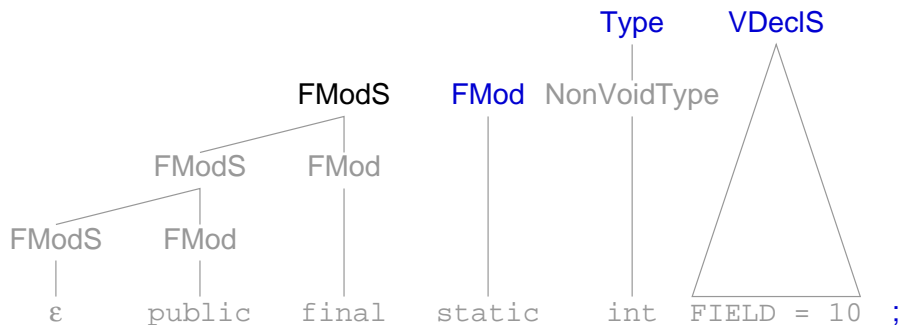
$\$ \langle FModS \rangle \langle FMod \rangle \parallel \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle ; \$$

$\stackrel{\text{reduce}}{=} \$ \parallel \langle FModS \rangle \langle FMod \rangle \langle Type \rangle \langle VDeclS \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



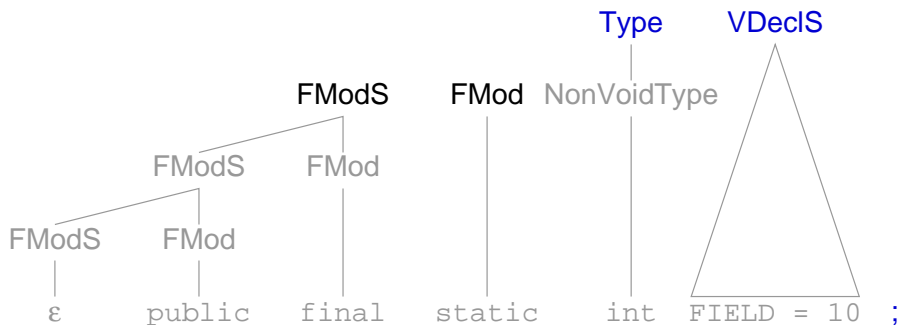
$\$ \parallel \langle FModS \rangle \quad \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

$\underset{\text{shift}}{\parallel} \$ \langle FModS \rangle \parallel \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



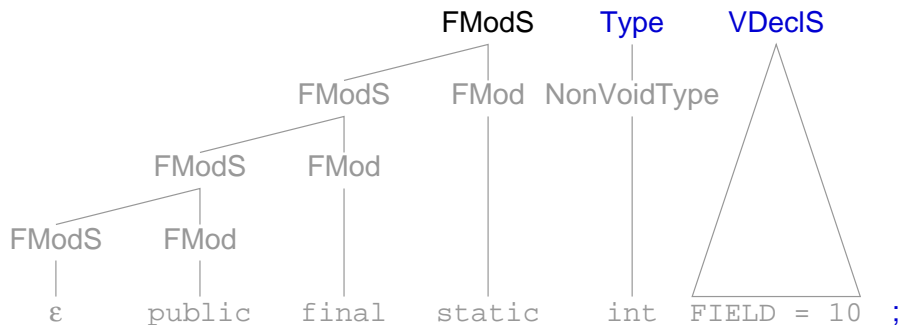
$\$ \langle FModS \rangle \parallel \langle FMod \rangle \langle Type \rangle \langle VDecls \rangle ; \$$

$\underset{\text{shift}}{\parallel} \$ \langle FModS \rangle \langle FMod \rangle \parallel \langle Type \rangle \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

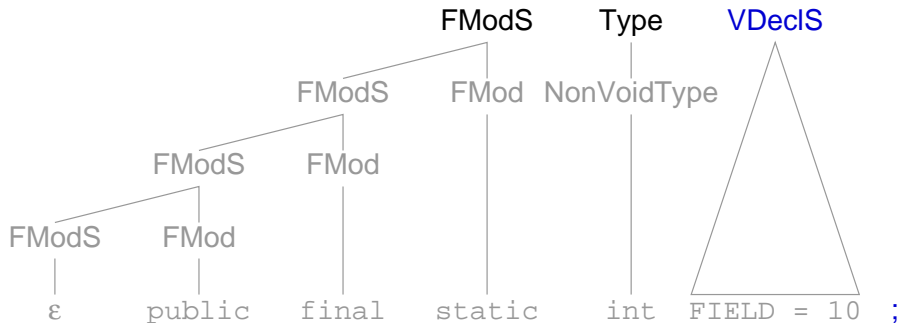


$\$ \parallel \langle FModS \rangle \langle Type \rangle \langle VDecls \rangle ; \$$
 $\underline{\underline{\text{shift}}} \$ \langle FModS \rangle \parallel \langle Type \rangle \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

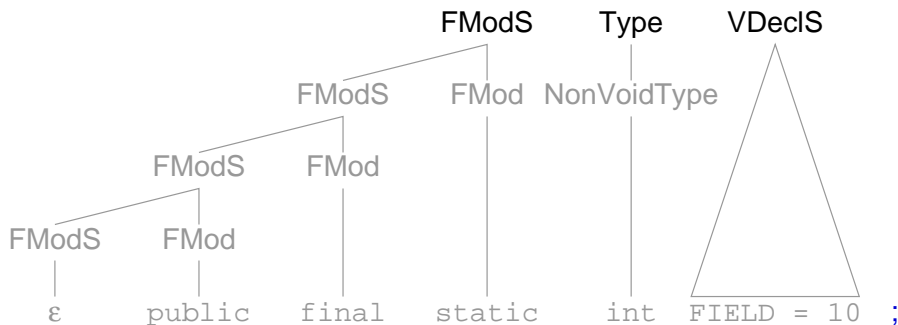


$\$ \langle FModS \rangle \parallel \langle Type \rangle \langle VDecls \rangle ; \$$
 $\underline{\text{shift}} \ \$ \langle FModS \rangle \langle Type \rangle \parallel \langle VDecls \rangle ; \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



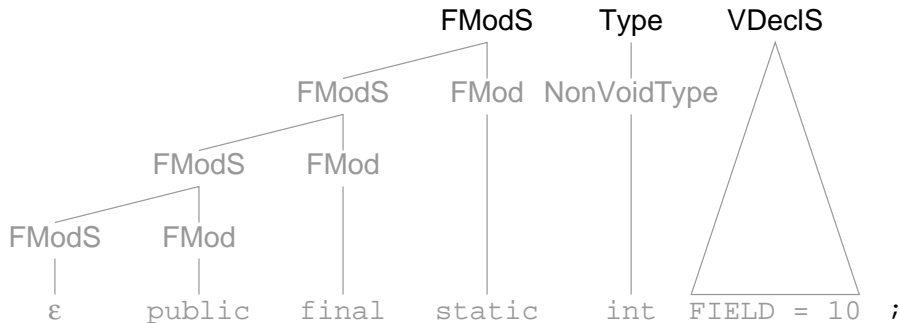
$$\$ \langle FModS \rangle \langle Type \rangle \parallel \langle VDeclS \rangle ; \$$$

$$\underline{\underline{\text{shift}}} \$ \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle \parallel ; \$$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example

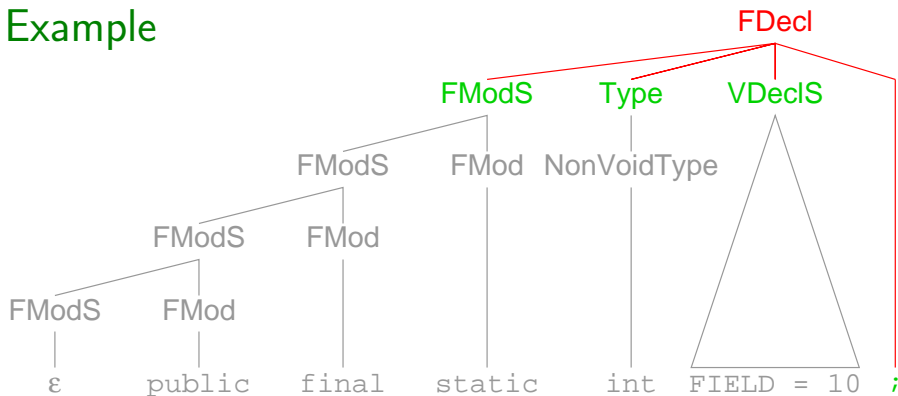


$$\begin{array}{l} \$ \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle \parallel ; \$ \\ \underline{\underline{\text{shift}}} \$ \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \parallel \$ \end{array}$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



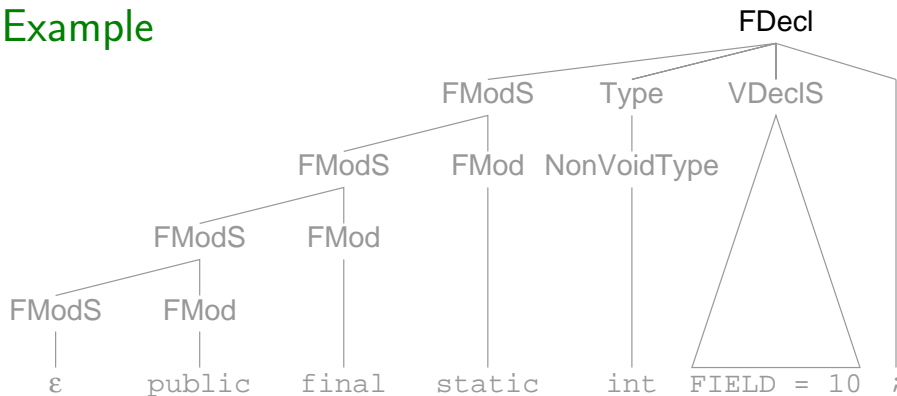
$\$ \langle FModS \rangle \langle Type \rangle \langle VDeclS \rangle ; \parallel \$$

$\stackrel{\text{reduce}}{\parallel} \$ \parallel \langle FDecl \rangle \$$

Noncanonical Parsing

Colmerauer [1970], Szymanski and Williams [1976], Tai [1979]

Example



\$ || *<FDecl>* \$

shift \$ *<FDecl>* || \$

Noncanonical Lookaheads

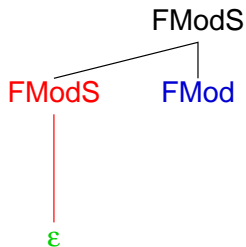
Example

FModS

ϵ

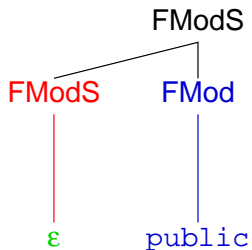
Noncanonical Lookaheads

Example



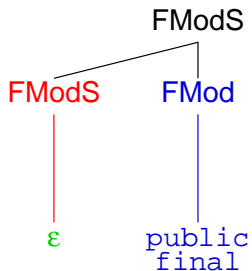
Noncanonical Lookaheads

Example



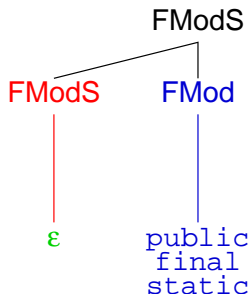
Noncanonical Lookaheads

Example



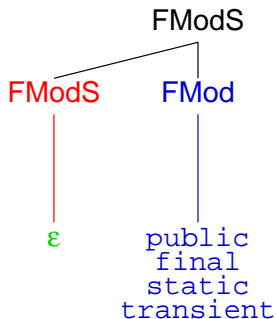
Noncanonical Lookaheads

Example



Noncanonical Lookaheads

Example



Noncanonical Lookaheads

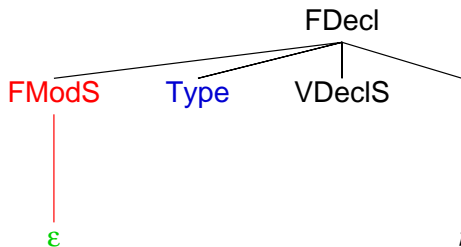
Example

FModS

ϵ

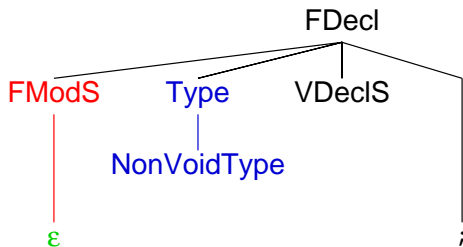
Noncanonical Lookaheads

Example



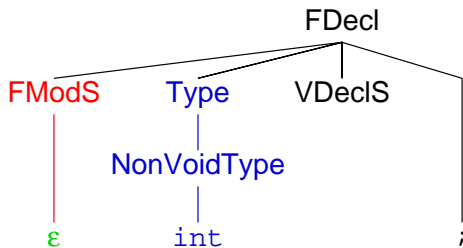
Noncanonical Lookaheads

Example



Noncanonical Lookaheads

Example



Noncanonical Lookaheads

Notation

$\langle FModS \rangle \rightarrow \varepsilon$

{public, final, static, int,
transient, $\langle FMod \rangle$, $\langle Type \rangle$,
 $\langle NonVoidType \rangle$ }

Noncanonical Lookaheads

Comparison

$\langle FModS \rangle \rightarrow \epsilon$ {public, final, static, int,
transient, $\langle FMod \rangle$, $\langle Type \rangle$,
 $\langle NonVoidType \rangle$ }

$\langle MModS \rangle \rightarrow \epsilon$ {public, final, static, int,
abstract, $\langle MMod \rangle$, $\langle ReturnType \rangle$,
 $\langle NonVoidType \rangle$ }

Noncanonical Lookaheads

Conflicts

$\langle FModS \rangle \rightarrow \varepsilon$ {public, final, static, int,
transient, $\langle FMod \rangle$, $\langle Type \rangle$,
 $\langle NonVoidType \rangle$ }

$\langle MModS \rangle \rightarrow \varepsilon$ {public, final, static, int,
abstract, $\langle MMod \rangle$, $\langle ReturnType \rangle$,
 $\langle NonVoidType \rangle$ }

Noncanonical Lookaheads

Decisions

$\langle FModS \rangle \rightarrow \varepsilon$ {public, final, static, int,
transient, $\langle FMod \rangle$, $\langle Type \rangle$,
 $\langle NonVoidType \rangle$ }

$\langle MModS \rangle \rightarrow \varepsilon$ {public, final, static, int,
abstract, $\langle MMod \rangle$, $\langle ReturnType \rangle$,
 $\langle NonVoidType \rangle$ }

Noncanonical Lookaheads

Reduced Lookaheads

$\langle FModS \rangle \rightarrow \epsilon$ {public, final, static, int, transient, $\langle FMod \rangle$, $\langle Type \rangle$, $\langle NonVoidType \rangle$ }

$\langle MModS \rangle \rightarrow \epsilon$ {public, final, static, int, abstract, $\langle MMod \rangle$, $\langle ReturnType \rangle$, $\langle NonVoidType \rangle$ }

Noncanonical Parsers

- ▶ Same parsing algorithm
- ▶ Different constructions

Noncanonical LALR(1)

- ▶ Generic construction
- ▶ From the canonical states collection
 - ▶ Noncanonical lookahead
 - ▶ Noncanonical states

Noncanonical LALR(1)

- ▶ Generic construction
- ▶ From the canonical states collection
 - ▶ Noncanonical lookahead
 - ▶ Noncanonical states

Noncanonical LALR(1)

- ▶ Generic construction
- ▶ From the canonical states collection
 - ▶ Noncanonical lookahead
 - ▶ Noncanonical states

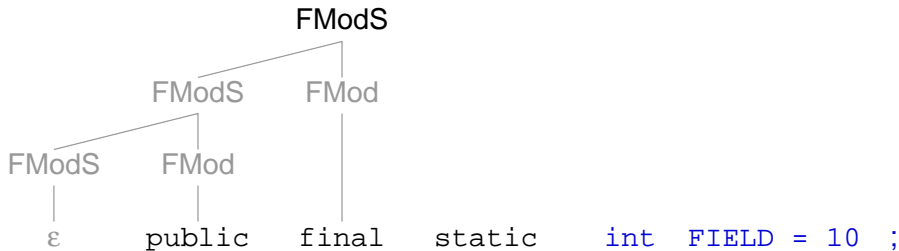
Valid Covers

Example

```
public final static int FIELD = 10 ;
```

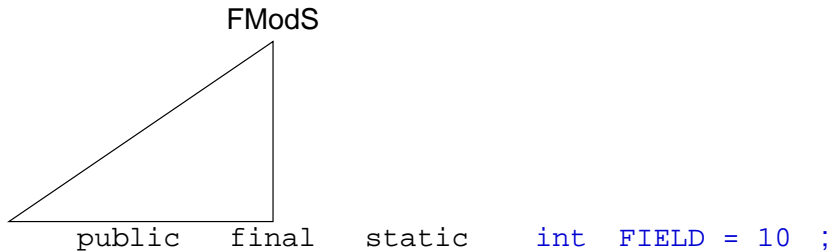
Valid Covers

Example



Valid Covers

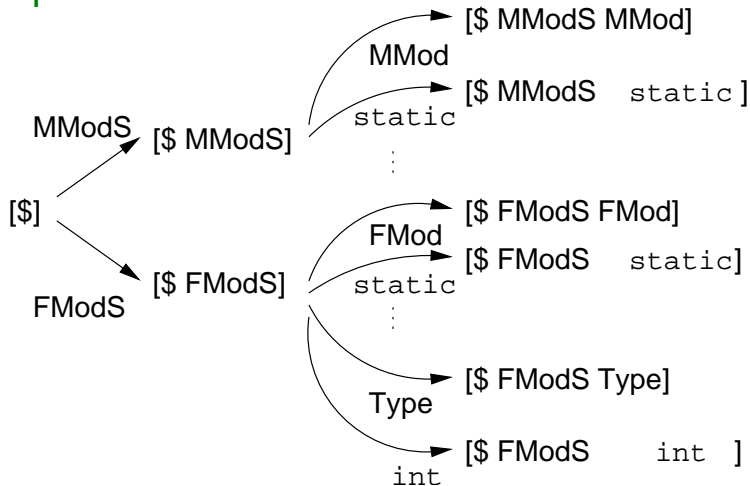
Example



Noncanonical Lookaheads

DeRemer and Pennello [1982]

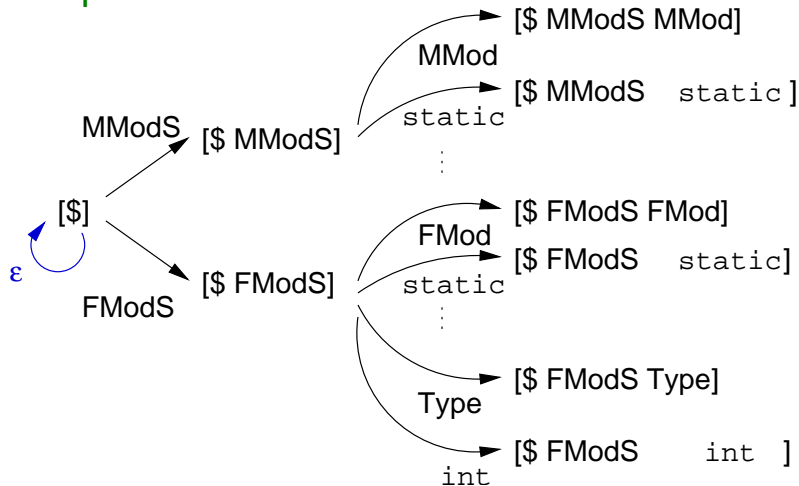
Example



Noncanonical Lookaheads

DeRemer and Pennello [1982]

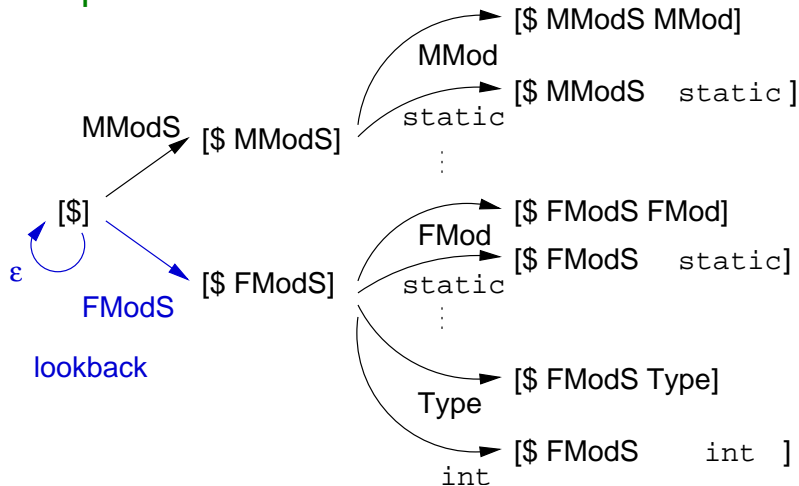
Example



Noncanonical Lookaheads

DeRemer and Pennello [1982]

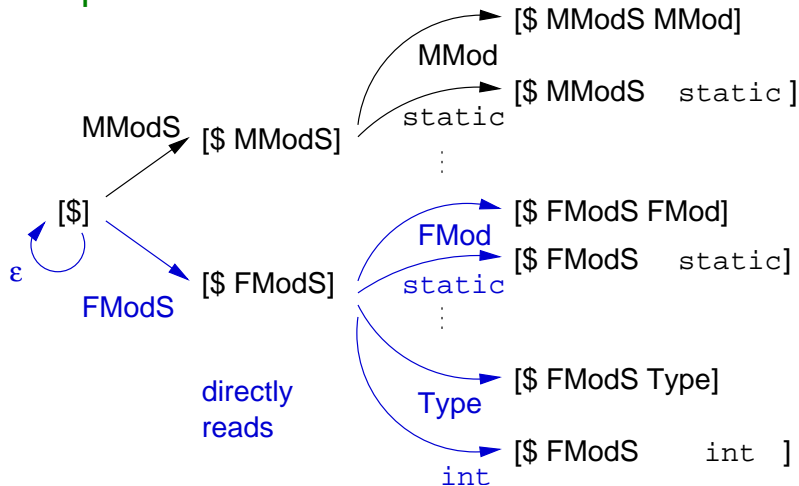
Example



Noncanonical Lookaheads

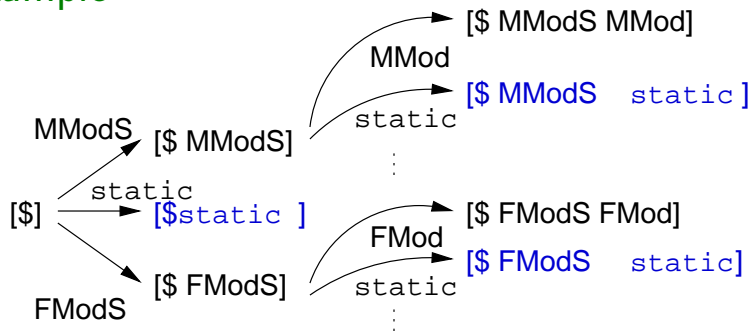
DeRemer and Pennello [1982]

Example



Noncanonical States

Example



$$\llbracket \$ \text{ static} \rrbracket = \{ \llbracket \$ \text{ static} \rrbracket, \llbracket \$ \langle FModS \rangle \text{ static} \rrbracket, \llbracket \$ \langle MModS \rangle \text{ static} \rrbracket \}$$

Closing Comments

Noncanonical Parsing

- ▶ Determinism
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity
- ▶ Robust parsing
- ▶ Incremental parsing
- ▶ Parallel parsing

Closing Comments

Noncanonical Parsing

- ▶ Determinism
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity
- ▶ Robust parsing
- ▶ Incremental parsing
- ▶ Parallel parsing

Closing Comments

Noncanonical Parsing

- ▶ Determinism
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity
- ▶ Robust parsing
- ▶ Incremental parsing
- ▶ Parallel parsing

Closing Comments

Noncanonical Parsing

- ▶ Determinism
- ▶ Parsing power
- ▶ $\mathcal{O}(n)$ time complexity
- ▶ Robust parsing
- ▶ Incremental parsing
- ▶ Parallel parsing

Closing Comments

NLALR(1)

- ▶ Simple & fast computations on canonical automaton
- ▶ Generic construction

Improvements

Better parsing power: no preset lookahead length

Syntactic Predicates

Parr and Quong [1995]

- ▶ User defined lookahead pattern

Example

```
((MMod)* ResType MDecl)?
```

- ▶ Error-prone
- ▶ Exponential time complexity

Syntactic Predicates

Parr and Quong [1995]

- ▶ User defined lookahead pattern

Example

`((MMod)* ResType MDecl)?`

- ▶ Error-prone
- ▶ Exponential time complexity

Syntactic Predicates

Parr and Quong [1995]

- ▶ User defined lookahead pattern

Example

`((MMod)* ResType MDecl)?`

- ▶ Error-prone
- ▶ Exponential time complexity

Syntactic Predicates

Parr and Quong [1995]

- ▶ User defined lookahead pattern

Example

$((MMod)^* ResType MDecl)?$

- ▶ Error-prone
- ▶ Exponential time complexity

LR-Regular Parsing

Baker [1981], Bermudez and Schimpf [1990], Farré and Fortes Gálvez [2001]

- ▶ Finite State lookahead discrimination

▶ $\mathcal{O}(n^2)$ time complexity

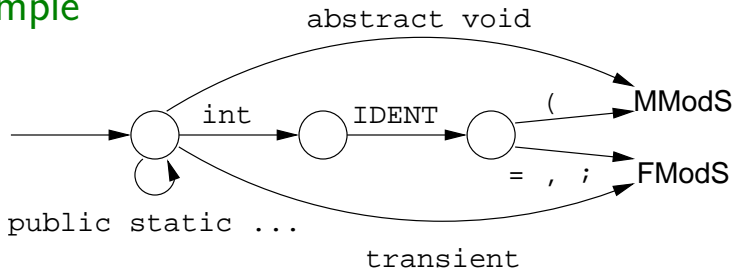
▶ Only considers terminal symbols

LR-Regular Parsing

Baker [1981], Bermudez and Schimpf [1990], Farré and Fortes Gálvez [2001]

- ▶ Finite State lookahead discrimination

Example



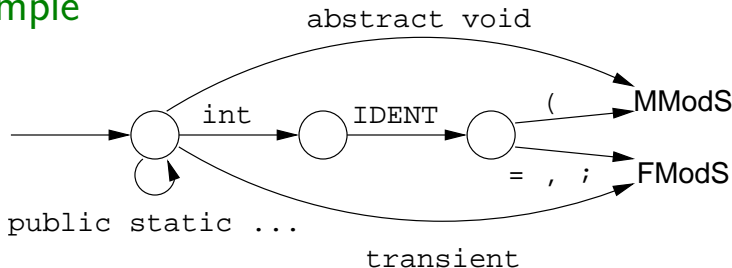
- ▶ $\mathcal{O}(n^2)$ time complexity
- ▶ Only considers terminal symbols

LR-Regular Parsing

Baker [1981], Bermudez and Schimpf [1990], Farré and Fortes Gálvez [2001]

- ▶ Finite State lookahead discrimination

Example



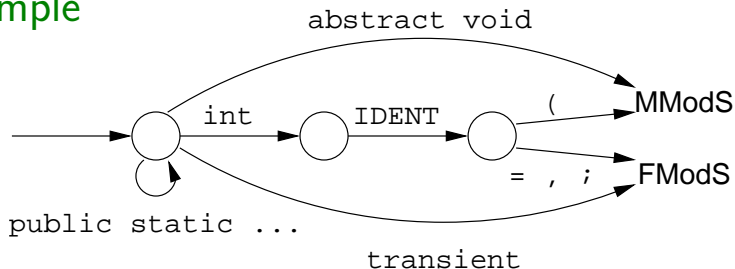
- ▶ $O(n^2)$ time complexity
- ▶ Only considers terminal symbols

LR-Regular Parsing

Baker [1981], Bermudez and Schimpf [1990], Farré and Fortes Gálvez [2001]

- ▶ Finite State lookahead discrimination

Example



- ▶ $\mathcal{O}(n^2)$ time complexity
- ▶ Only considers terminal symbols

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

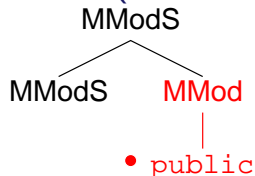
$MModS$

$MModS \bullet MMod$

$\langle MModS \rangle \rightarrow \langle MModS \rangle \bullet \langle MMod \rangle$

LR(0) Construction

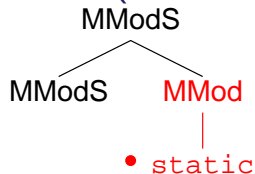
$\text{Valid}_0(\$ \langle MModS \rangle)$



$\langle MMod \rangle \rightarrow \cdot \text{public}$

LR(0) Construction

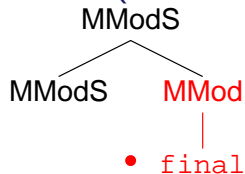
$\text{Valid}_0(\$ \langle MModS \rangle)$



$\langle MMod \rangle \rightarrow \bullet static$

LR(0) Construction

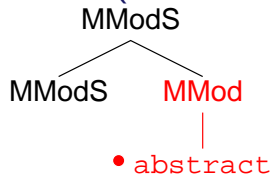
$\text{Valid}_0(\$ \langle MModS \rangle)$



$\langle MMod \rangle \rightarrow \cdot \text{final}$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$



$\langle MMod \rangle \rightarrow \cdot \text{abstract}$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

MHead

MModS • *ResType* *MDecl* *Throws*

$\langle MHead \rangle \rightarrow \langle MModS \rangle \cdot \langle ResType \rangle \langle MDecl \rangle \langle Thro$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

MHead

MModS

ResType

MDecl

Throws

• void

$\langle ResType \rangle \rightarrow \cdot \langle NonVoidType \rangle$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

MHead

MModS

ResType

MDecl

Throws

• NonVoidType

$\langle ResType \rangle \rightarrow \cdot \text{void}$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

MHead

MModS ResType MDecl Throws

NonVoidType

• int

$\langle NonVoidType \rangle \rightarrow \cdot int$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$

$\langle MModS \rangle \rightarrow \langle MModS \rangle . \langle MMod \rangle$

$\langle MMod \rangle \rightarrow .\text{public}$

$\langle MMod \rangle \rightarrow .\text{static}$

$\langle MMod \rangle \rightarrow .\text{final}$

$\langle MMod \rangle \rightarrow .\text{abstract}$

$\langle MHead \rangle \rightarrow \langle MModS \rangle . \langle ResType \rangle \langle MDecl \rangle$

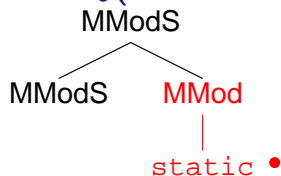
$\langle ResType \rangle \rightarrow . \langle NonVoidType \rangle$

$\langle ResType \rangle \rightarrow .\text{void}$

$\langle NonVoidType \rangle \rightarrow .\text{int}$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle \text{ static})$



$\langle MMod \rangle \rightarrow \text{static}.$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle \langle MMod \rangle)$

$MModS$

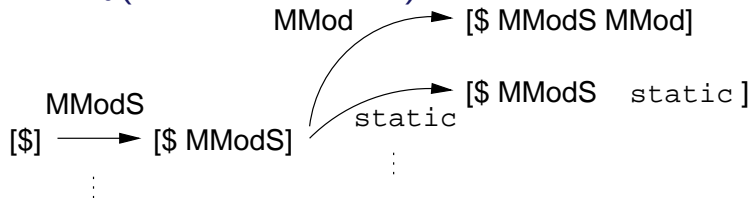
$MModS$

$MMod \bullet$

$\langle MModS \rangle \rightarrow \langle MModS \rangle \langle MMod \rangle \bullet$

LR(0) Construction

$\text{Valid}_0(\$ \langle MModS \rangle)$



- T. P. Baker. Extending lookahead for LR parsers. *J. Comput. Syst. Sci.*, 22(2):243–259, 1981. doi: 10.1016/0022-0000(81)90030-1.
- M. E. Bermudez and K. M. Schimpf. Practical arbitrary lookahead LR parsing. *J. Comput. Syst. Sci.*, 41(2): 230–250, 1990. doi: 10.1016/0022-0000(90)90037-L.
- A. Colmerauer. Total precedence relations. *J. ACM*, 17 (1):14–30, Jan. 1970. doi: 10.1145/321556.321559.
- F. DeRemer and T. Pennello. Efficient computation of LALR(1) look-ahead sets. *ACM Trans. Prog. Lang. Syst.*, 4(4):615–649, 1982. doi: 10.1145/69622.357187.
- J. Farré and J. Fortes Gálvez. A bounded-connect construction for LR-regular parsers. In R. Wilhelm, editor, *CC'01*, volume 2027 of *Lecture Notes in*

- J. Gosling, B. Joy, and G. Steele. **The Java™ Language Specification**. Addison-Wesley Longman Publishing Co., Inc., first edition, Aug. 1996. ISBN 0-201-63451-1. URL <http://java.sun.com/docs/books/jls/>.
- D. E. Knuth. On the translation of languages from left to right. **Information and Control**, 8:607–639, 1965.
- T. J. Parr and R. W. Quong. ANTLR: A predicated-LL(k) parser generator. **Software, Practice and Experience**, 25(7):789–810, 1995. ISSN 0038-0644. URL <http://citeseer.nj.nec.com/12770>.

- S. Schmitz. Noncanonical LALR(1) parsing. In Z. Dang and O. H. Ibarra, editors, **DLT'06**, volume 4036 of **Lecture Notes in Computer Science**, pages 95–107. Springer, 2006.
- T. G. Szymanski and J. H. Williams. Noncanonical extensions of bottom-up parsing techniques. **SIAM J. Comput.**, 5(2):231–250, June 1976. URL http://locus.siam.org/SICOMP/volume-05/art_0205019.html.
- K.-C. Tai. Noncanonical SLR(1) grammars. **ACM Trans. Prog. Lang. Syst.**, 1(2):295–320, 1979. ISSN 0164-0925. doi: 10.1145/357073.357083.