Language Processing

(SYNTAX)

LANGUAGE PROCESSING

Language processing involves two distinct aspects :

☐ **Syntax (the structure of expressions) :**

- Formal definition of syntax
- Derivation of expressions
- Syntax trees
- Ambiguity
- Parsing

☐ **Semantics (the meaning of expressions) :**

- Next week

FORMAL DEFINITION OF SYNTAX

```
sentence ::= proper_noun intransitive_verb

proper_noun       ::= john
                    | mary


intransitive_verb ::= walks
                    | runs
                    |sleeps
```

The language defined by this grammar is :

```
{john walks, john runs, john sleeps,
 mary walks, mary runs, mary sleeps}
```

DERIVATIONS

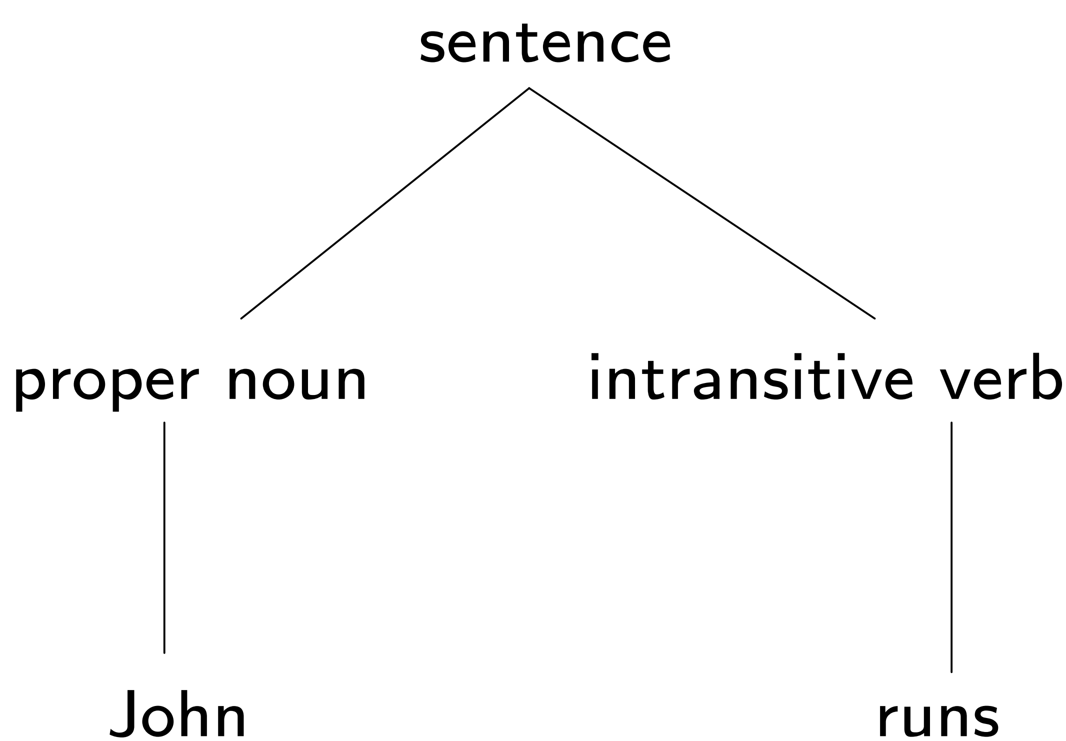
- An example of a derivation :

```
sentence => proper_noun intransitive_verb


proper_noun intransitive_verb
                 => john intransitive_verb


john intransitive_verb => john runs
```

Syntax Trees



PARSING

- Parsing is the process by which a syntax tree is constructed, eg top down parsing.
- Example, parse "john runs" with earlier grammar :

```
build root :        sentence

expand root :       sentence

         proper_noun  intransitive_verb

expand leaf :       sentence

         proper_noun    intransitive_verb

            john            ETC.
```

Formal Definition of a Grammar

A Grammar G is 4-tuple $G(T,N,S,P)$

where

T	is the set of terminal symbols
N	is the set of non-terminal symbols
S	is the distinguished non-terminal symbol
P	is the set of production rules

A Simple Grammar

$$T = \{0,1,2,3,4,5,6,7,8,9,.,-\}$$

$$N = \{digit, number\}$$

$$S = \{number\}$$

$$P = \{ \quad number ::= digit$$

$$number ::= digit\ number$$

$$number ::= number\ .\ number$$

$$number ::= -\ number$$

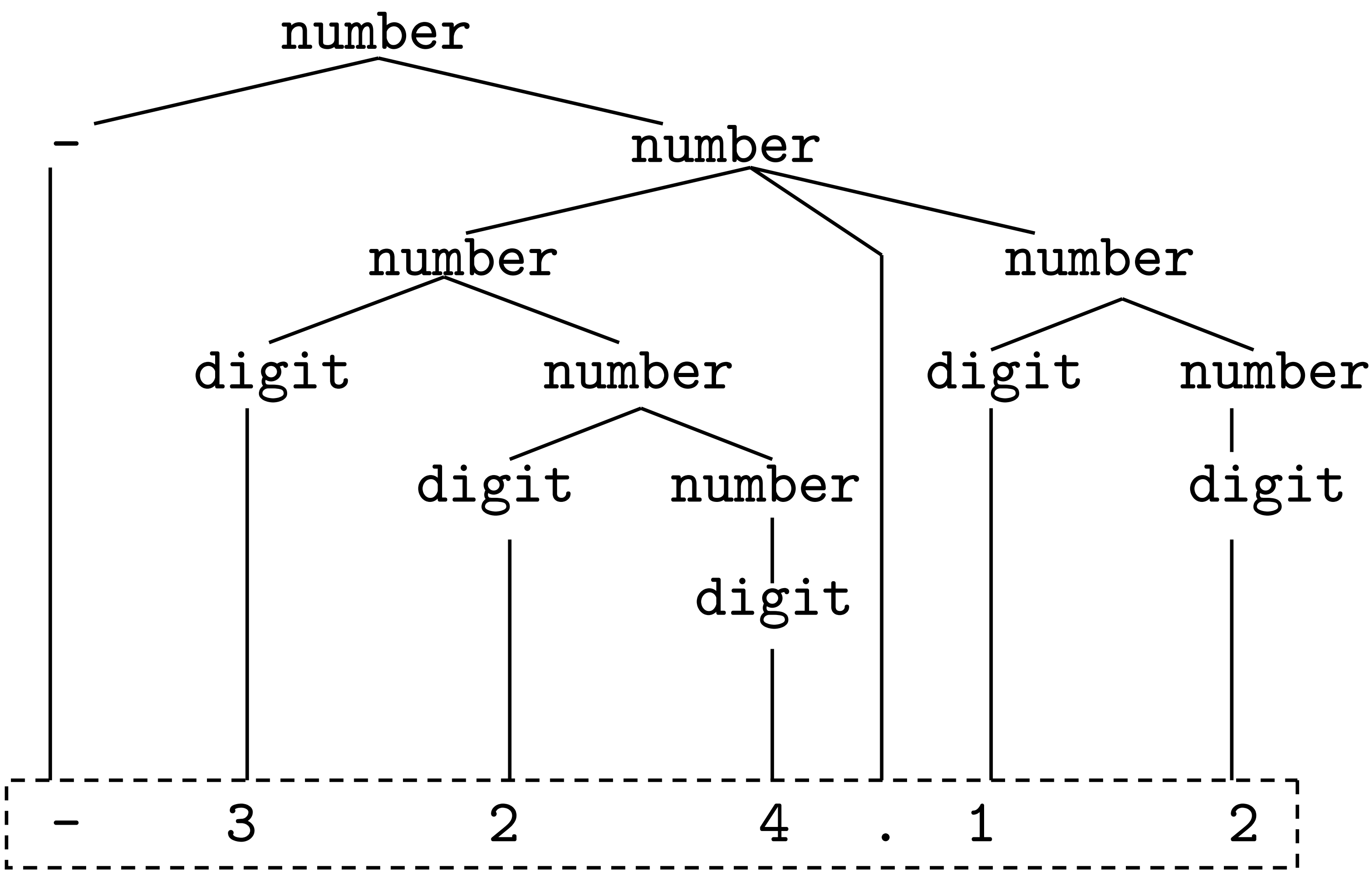
$$digit ::= 0|1|2|3|4|5|6|7|8|9$$

$$\}$$

Derivation of ’-324.12’

```
number
⇒ - number
⇒ - number . number
⇒ - digit number . number
⇒ - 3 number . number
⇒ - 3 digit number . number
⇒ - 3 2 number . number
⇒ - 3 2 digit . number
⇒ - 3 2 4 . number
⇒ - 3 2 4 . digit number
⇒ - 3 2 4 . 1 number
⇒ - 3 2 4 . 1 digit
⇒ - 3 2 4 . 1 2
```

Syntax Tree


number
-
number
number
number
digit
number
digit
number
digit
number
digit
digit
- 3 2 4 . 1 2

A Grammar for Arithmetic Expressions

$G(T,N,S,P)$

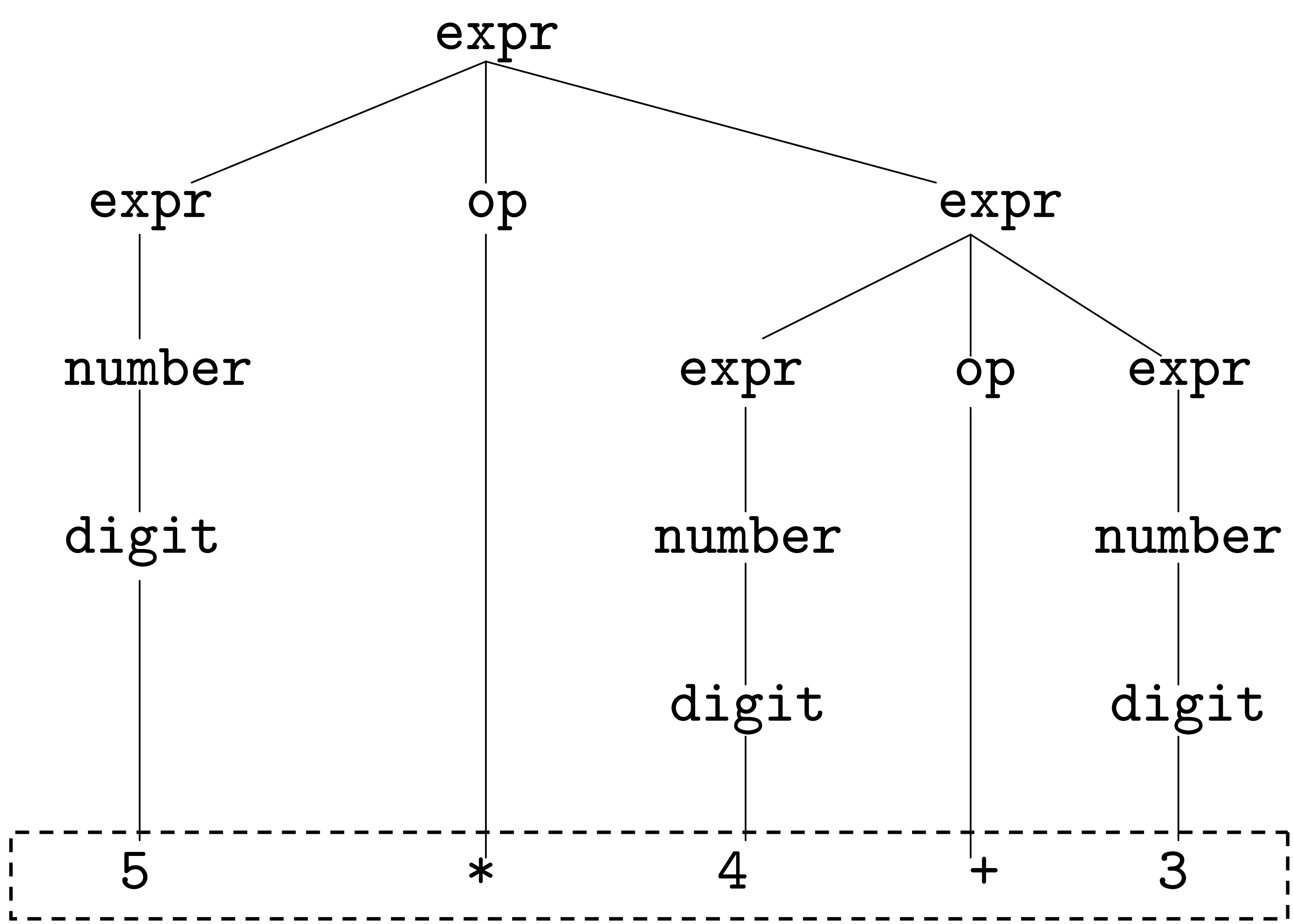
$$T = \{0,1,2,3,4,5,6,7,8,9,+,*,(,)\}$$

$$N = \{digit, op, expr, number\}$$

$$S = \{expr\}$$

$$\begin{aligned} P = \{ \quad & expr ::= number, \\ & expr ::= (expr), \\ & expr ::= expr\ op\ expr, \\ & op ::= + | *, \\ & number ::= digit \\ & number ::= digit\ number \\ & digit ::= 0|1|2|3|4|5|6|7|8|9 \\ \} & \end{aligned}$$

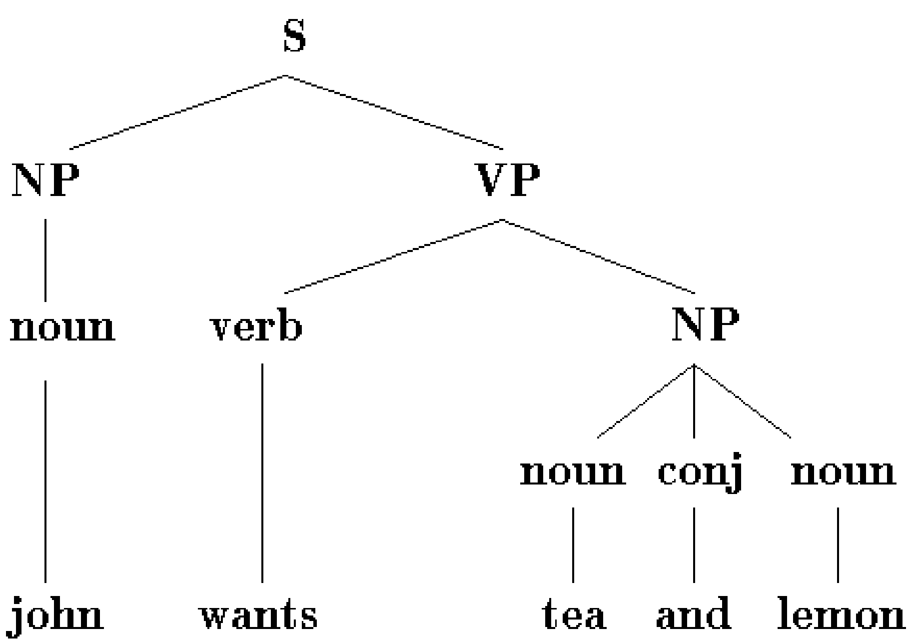
Syntax Trees

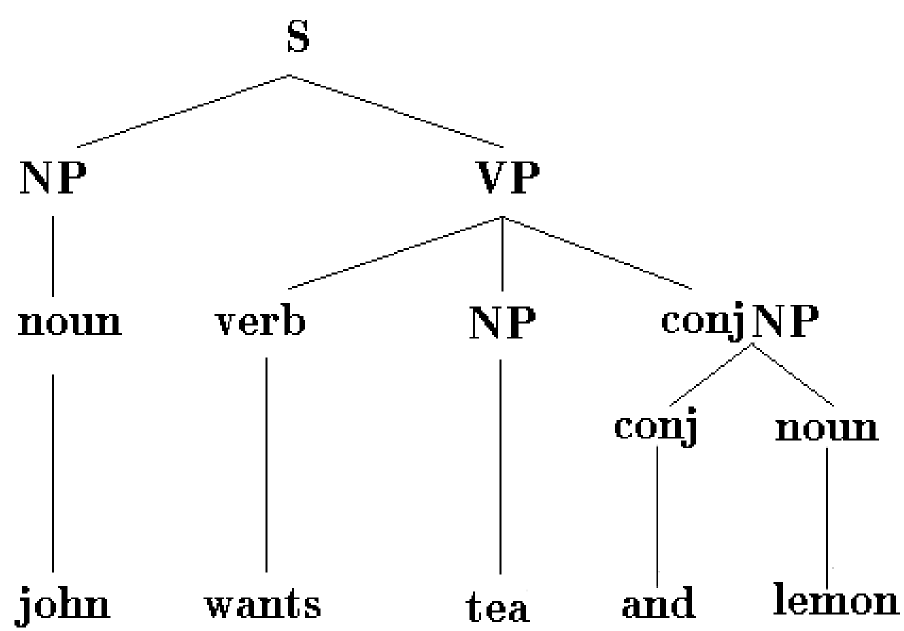


Using Grammars in Derivations

```
expr ::= expr op expr
⇒ number op expr
⇒ digit op expr
⇒ 5 op expr
⇒ 5 * expr
⇒ 5 * expr op expr
⇒ 5 * number op expr
⇒ 5 * digit op expr
⇒ 5 * 4 op expr
⇒ 5 * 4 + expr
⇒ 5 * 4 + number
⇒ 5 * 4 + digit
⇒ 5 * 4 + 3
```

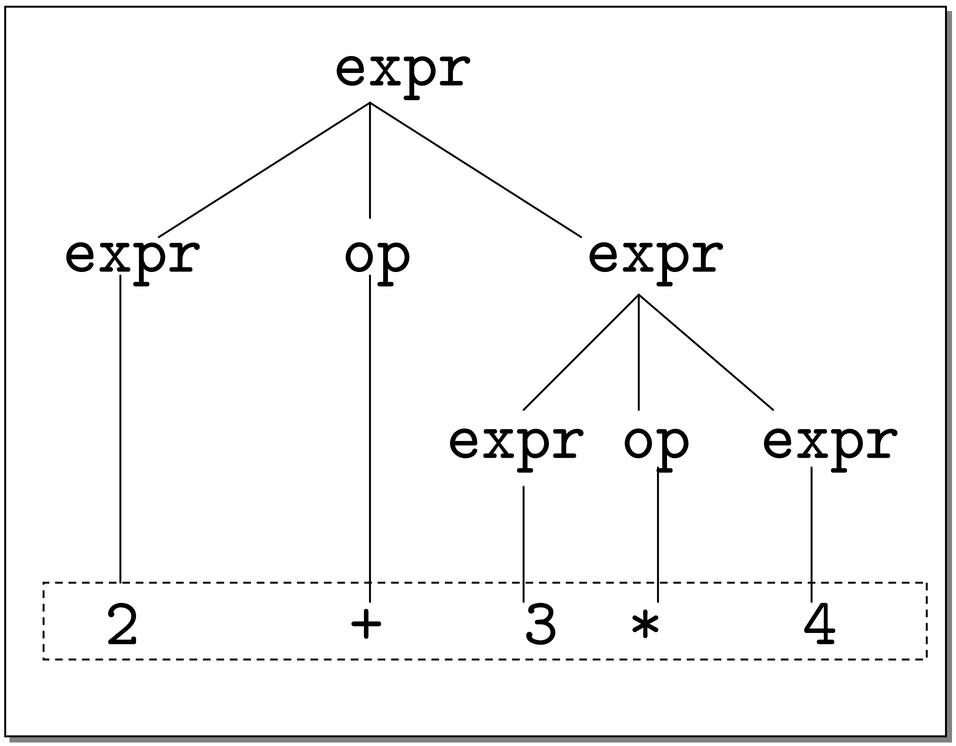
Ambiguous Grammars (Syntactic Ambiguity)

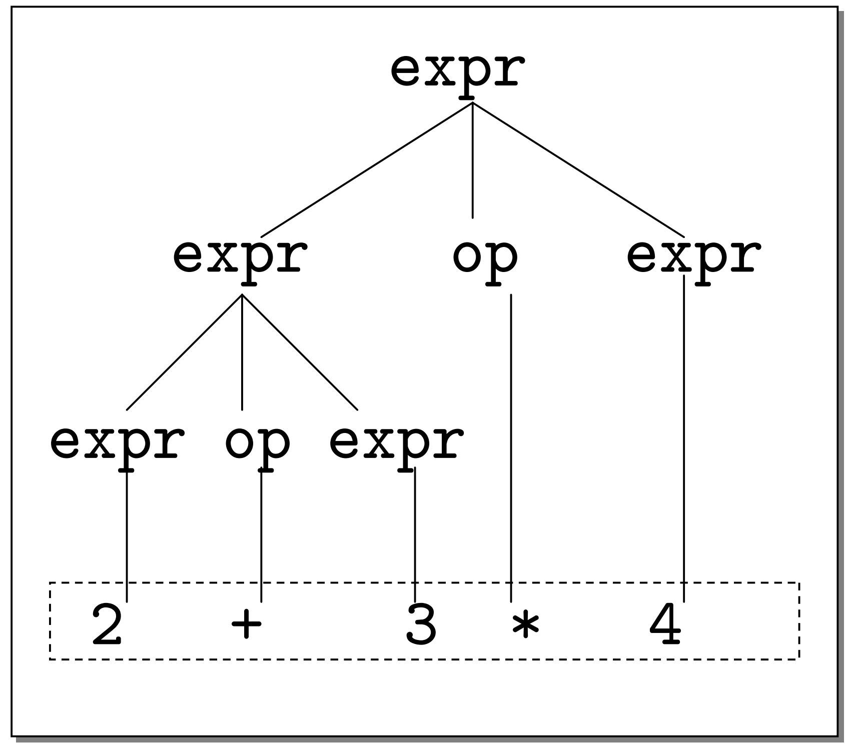




An Example of an Ambiguous Grammar

```
expr ::= 1|2|3|4|...            op ::= *|+
       | expr op expr
```





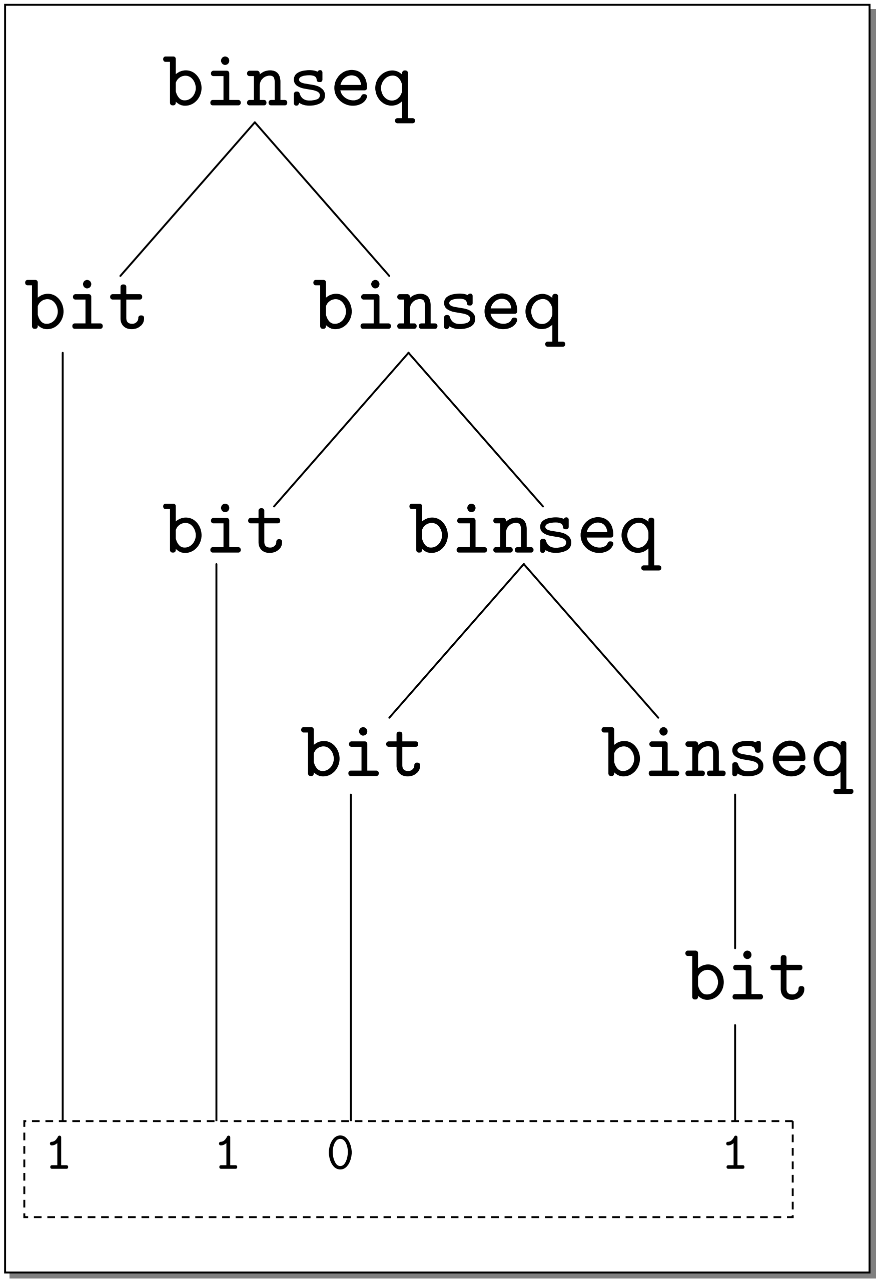
Right and Left Recursive Grammars

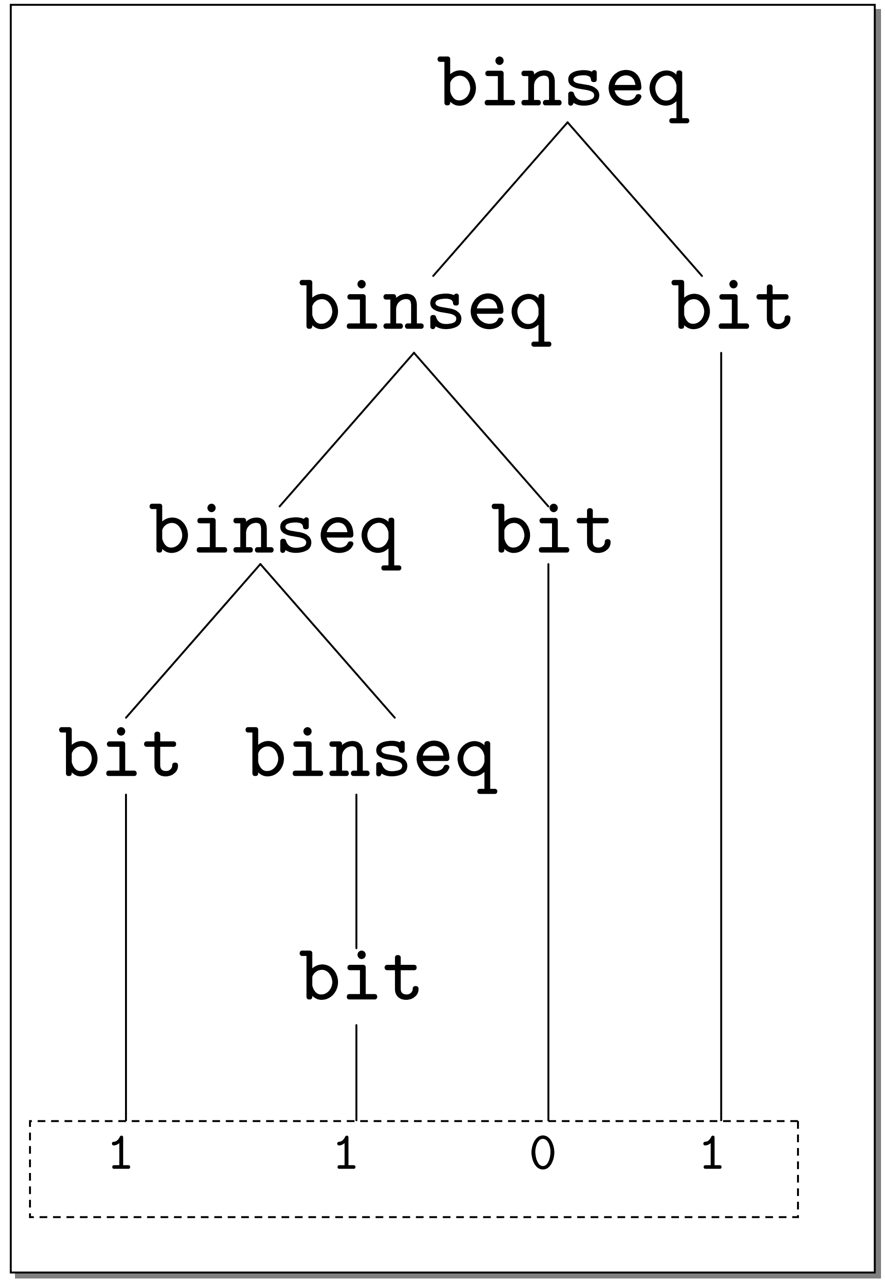
```
bit ::= 0|1
```

```
binseq ::=  bit
         |  bit binseq
```

```
binseq ::=  bit
         |  binseq bit
```

Right and Left Recursive Grammars





Context Sensitive Grammars

Can we define a grammar for palindromes?

```
a
aba
aabaa
aabbbaaa

pal  ::=  null
       |  char
       |  char pal char


does this work?
```

Semantic Rules (example: type checking)

Example valid lists are the following:

[1,2,3,4]

[ ]

[“john”, “steve”, “mary”, ...]

[[1,2,3],[1,2,34]]

[[“john”, “steve”],[“mary”]]

A Grammar for Web Pages (HTML)

```
htmlDoc ::= <html> head body </html>

head ::= <head> metaName title </head>
body ::= <body> contentObjs </body>

contentObjs ::= contentObj
              | contentObj contentObjs

contentObj  ::= parag
             | list
             | anchor
             | image
             | applet
```

Defining a Grammar for a Language

Specify a grammar that defines the language of propositional logic. These are examples of valid expressions (wff) in propositional logic:

$$((p \wedge q) \rightarrow r) \leftrightarrow \neg r$$

$$p \rightarrow q$$

$$q$$

$$\neg p \vee q$$

$$(p \wedge q) \rightarrow r$$

Define the Language of First-Order Predicate Logic

$(\forall x)(P(x) \rightarrow Q(x))$

$P(x)$

$P(john)$

$R(john, marry)$

$(\forall y)(\exists x)((P(x) \wedge S(y)) \rightarrow Q(x))$

$(\exists x)(\neg P(x) \vee Q(x))$

The Language of FOPL

$$
\begin{aligned}
wff &::= predicate \\
&\quad \mid wff \quad op \quad wff \\
&\quad \mid (wff) \\
&\quad \mid \neg wff \\
&\quad \mid (quant \quad var) \quad wff
\end{aligned}
$$

$op ::= \vee \mid \wedge \mid \rightarrow \mid \leftrightarrow$

$quant ::= \exists \mid \forall$

$predicate ::= symbol \quad (\quad sequence \quad)$

$term ::= const \mid var$

$sequence ::= term \mid term \quad , \quad sequence$

$var ::= x \mid y \mid z \mid x_1 \mid y_1 \mid z_1 \mid etc\ldots$

$symbol ::= P \mid Q \mid R \mid S \mid P_1 \mid Q_1 \mid etc\ldots$

An Example *wff* in FOL

$(\forall x)(P(x) \rightarrow Q(x))$

$$
\begin{aligned}
\mathit{wff} &\Rightarrow (\mathit{quant}\ \ \mathit{var})\ \ \mathit{wff} \\
&\Rightarrow (\forall\ \ \mathit{var})\ \ \mathit{wff} \\
&\Rightarrow (\forall\ x)\ \ (\mathit{wff}) \\
&\Rightarrow (\forall\ x)\ \ (\mathit{wff}\quad \mathit{op}\quad \mathit{wff}) \\
&\Rightarrow (\forall\ x)\ \ (\mathit{wff}\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall\ x)\ \ (\mathit{predicate}\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall\ x)\ \ (\mathit{symbol}\,(\mathit{sequence})\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall x)\,(P(\mathit{sequence})\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall x)\,(P(\mathit{term})\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall x)\,(P(\mathit{var})\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall x)\,(P(x)\quad \rightarrow\quad \mathit{wff}) \\
&\Rightarrow (\forall x)\,(P(x)\quad \rightarrow\quad \mathit{symbol}\,(\mathit{sequence})) \\
&\Rightarrow (\forall x)\,(P(x)\ \rightarrow\ Q\,(\mathit{sequence})) \\
&\Rightarrow (\forall x)\,(P(x)\ \rightarrow\ Q(\mathit{term})) \\
&\Rightarrow (\forall x)\,(P(x)\ \rightarrow\ Q(\mathit{var})) \\
&\Rightarrow (\forall x)\,(P(x)\ \rightarrow\ Q(x))
\end{aligned}
$$