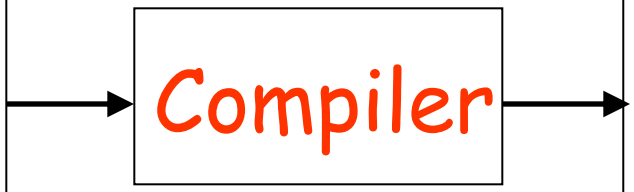
Compilers

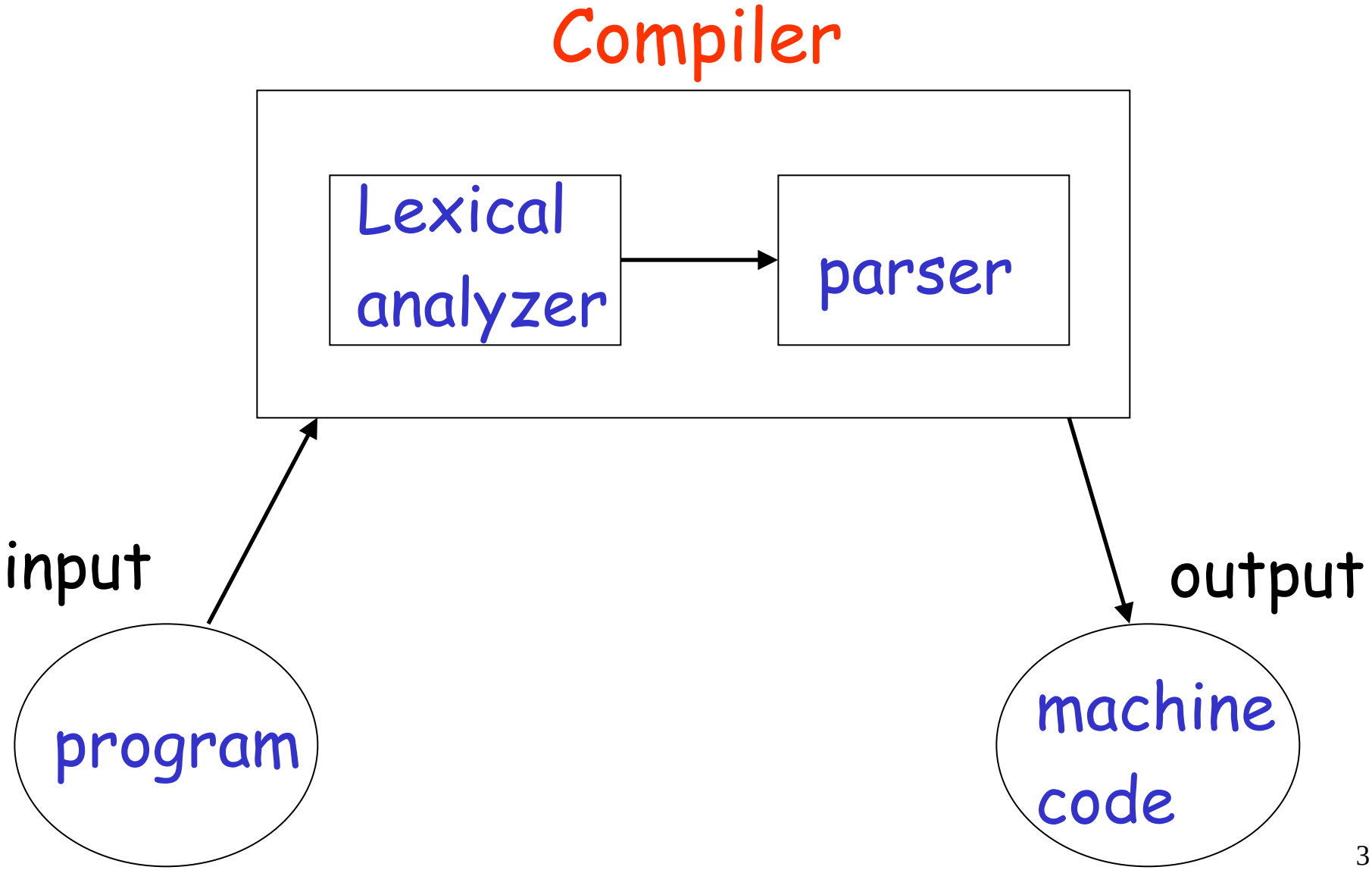
Program

```
v = 5;
if (v>5)
   x = 12 + v;
while (x !=3) {
  x = x - 3;
  v = 10;
}
.....
```



Machine Code

```
Add v,v,0
cmp v,5
jmplt ELSE
THEN:
  add x, 12,v
ELSE:
WHILE:
cmp x,3
...
```


Compiler
Lexical analyzer
parser
input
output
program
machine code

A parser knows the grammar of the programming language

Parser

PROGRAM → STMT_LIST

STMT_LIST → STMT; STMT_LIST | STMT;

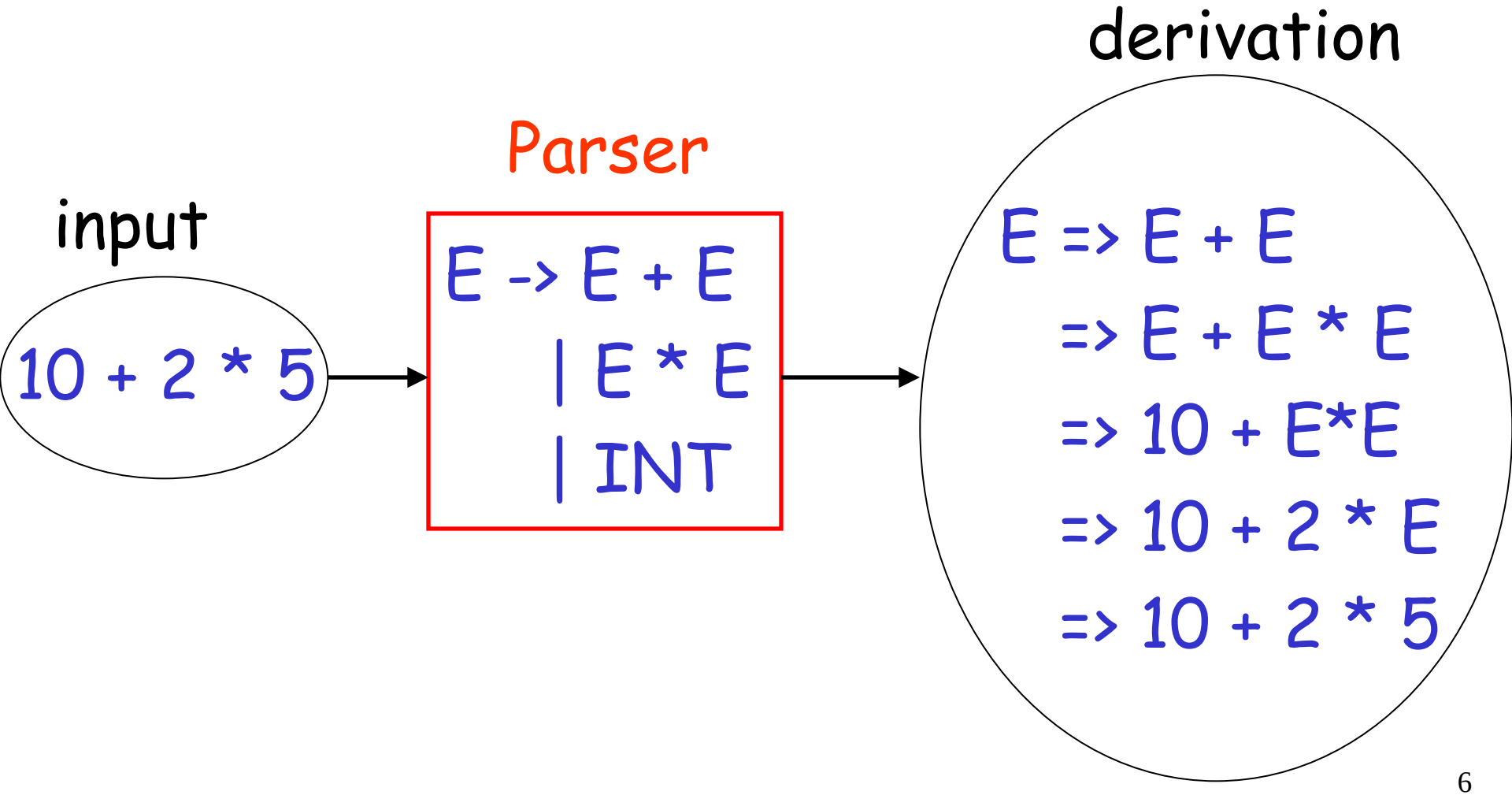
STMT → EXPR | IF_STMT | WHILE_STMT
| { STMT_LIST }

EXPR → EXPR + EXPR | EXPR - EXPR | ID

IF_STMT → if (EXPR) then STMT
| if (EXPR) then STMT else STMT

WHILE_STMT → while (EXPR) do STMT

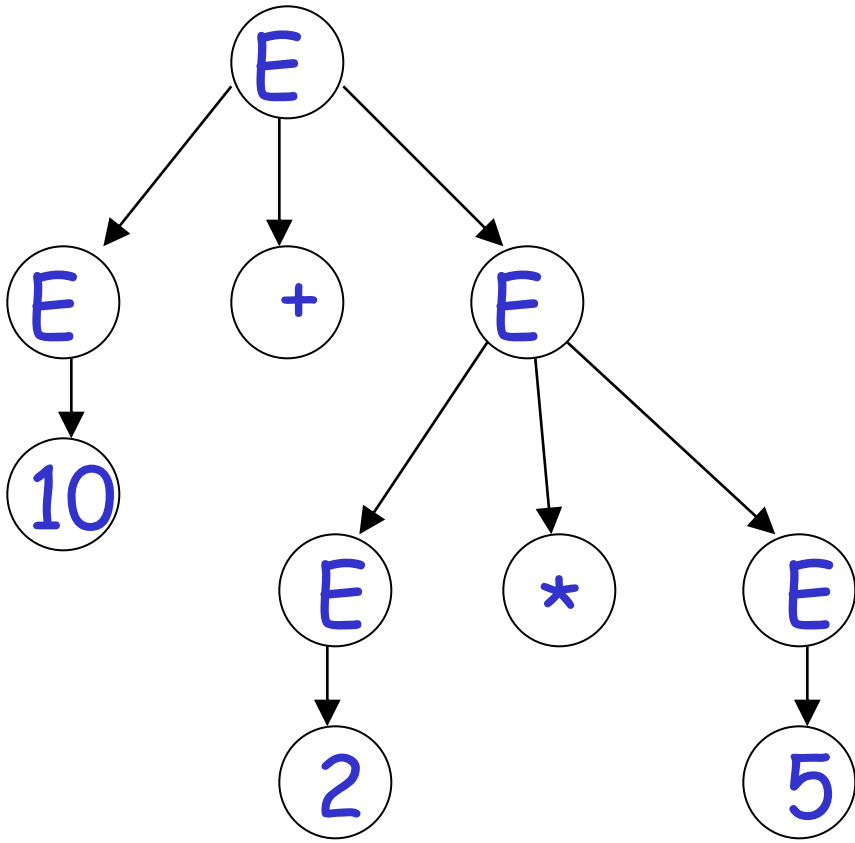
The parser finds the derivation of a particular input



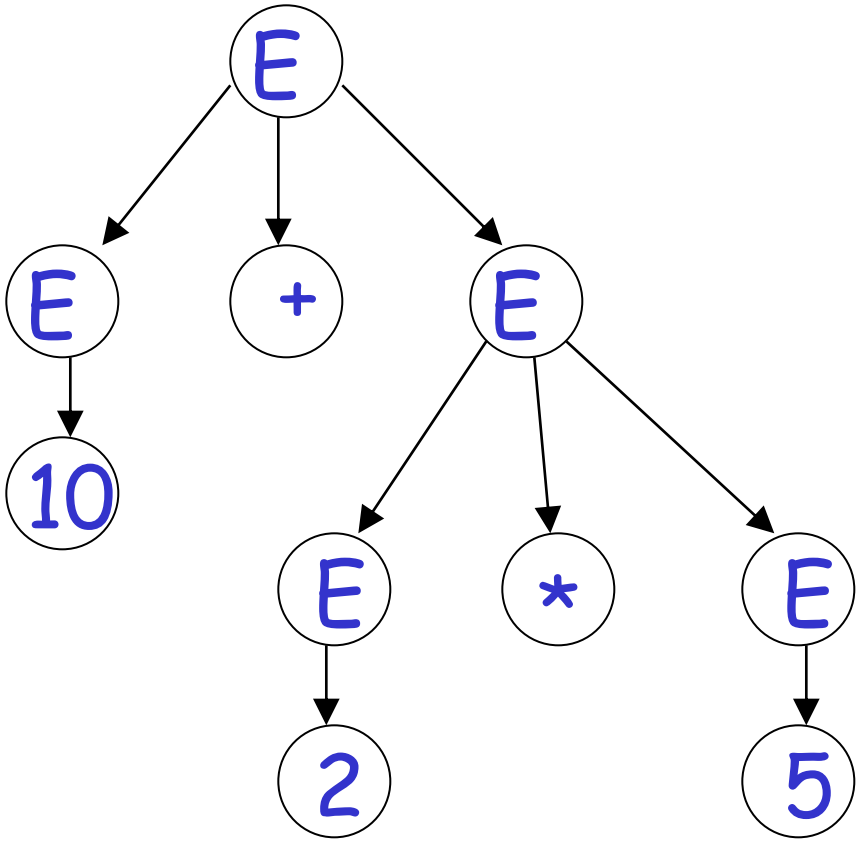
derivation

E => E + E
=> E + E * E
=> 10 + E*E
=> 10 + 2 * E
=> 10 + 2 * 5

derivation tree



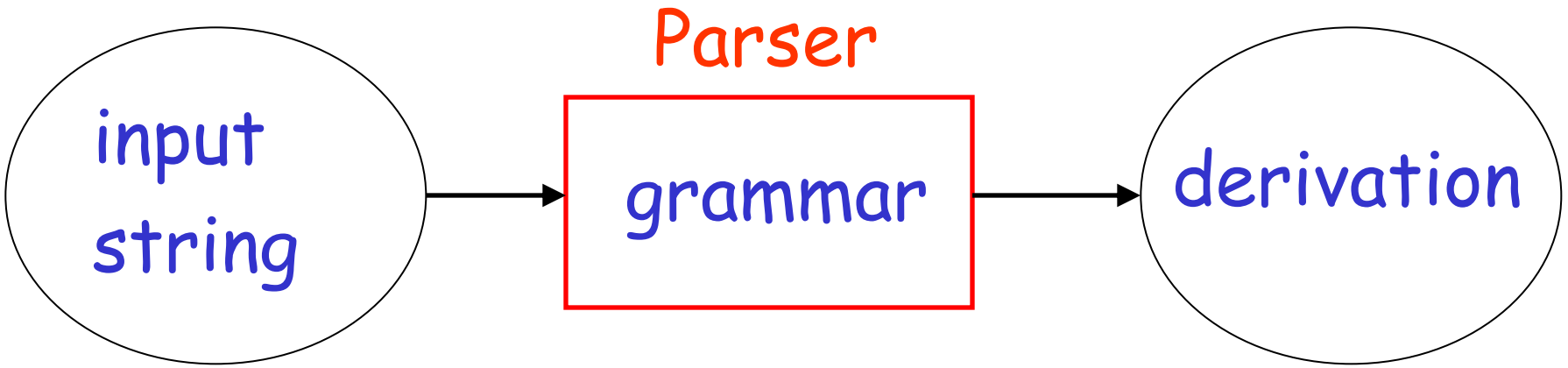
derivation tree



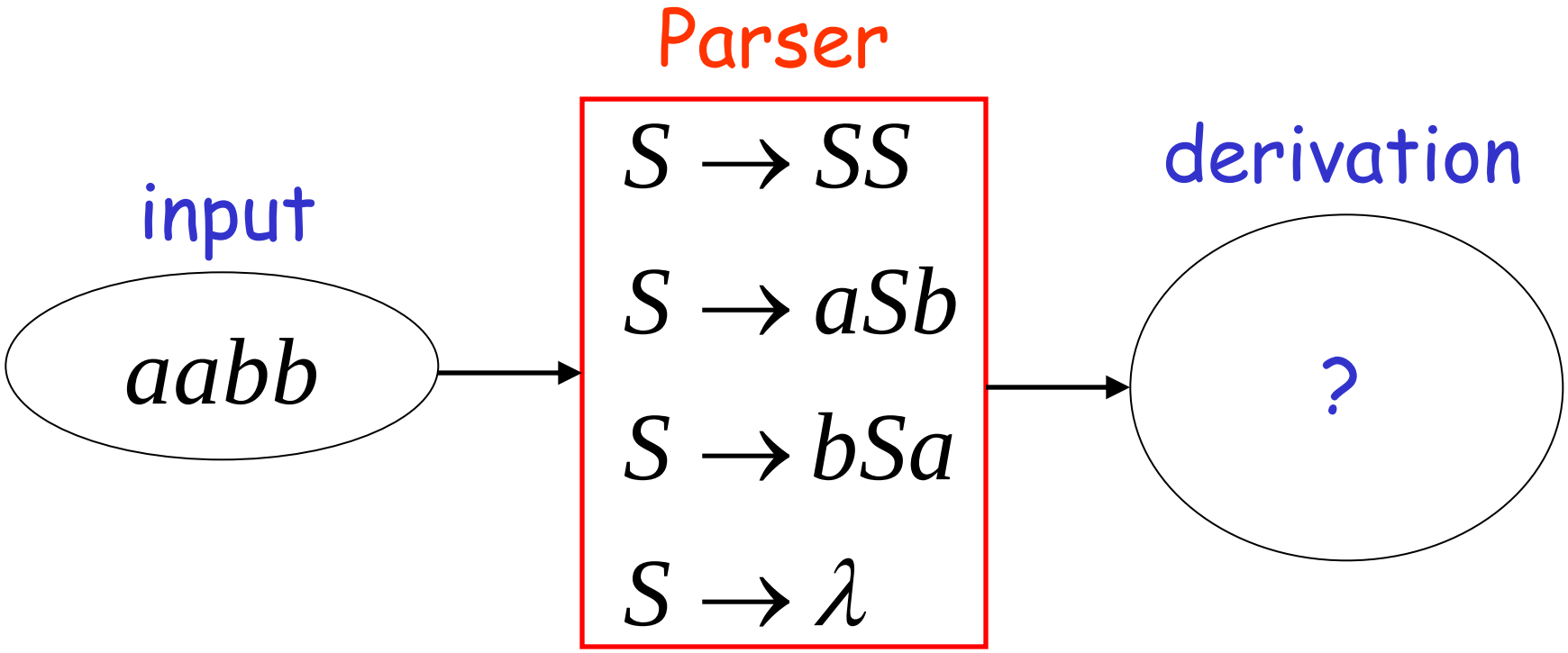
machine code

```
mult a, 2, 5
add b, 10, a
```

Parsing


Parser
input string
grammar
derivation

Example:


Parser
input
aabb
$S \to SS$
$S \to aSb$
$S \to bSa$
$S \to \lambda$
derivation
?

Exhaustive Search

$$S \to SS \mid aSb \mid bSa \mid \lambda$$

Phase 1:

$$S \Rightarrow SS$$

$$S \Rightarrow aSb$$

$$S \Rightarrow bSa$$

$$S \Rightarrow \lambda$$

Find derivation of

$$aabb$$

All possible derivations of length 1

$S \Rightarrow SS$

$S \Rightarrow aSb$

~~$S \Rightarrow bSa$~~

~~$S \Rightarrow \lambda$~~

$aabb$

Phase 2 $S \to SS \mid aSb \mid bSa \mid \lambda$

Phase 1

$S \Rightarrow SS$

$S \Rightarrow SS \Rightarrow SSS$

$S \Rightarrow SS \Rightarrow aSbS$

$S \Rightarrow SS \Rightarrow bSaS$

$S \Rightarrow SS \Rightarrow S$

$aabb$

$S \Rightarrow aSb$

$S \Rightarrow aSb \Rightarrow aSSb$

$S \Rightarrow aSb \Rightarrow aaSbb$

$S \Rightarrow aSb \Rightarrow abSab$

$S \Rightarrow aSb \Rightarrow ab$

$$S \to SS \mid aSb \mid bSa \mid \lambda$$

Phase 2

$$S \Rightarrow SS \Rightarrow SSS$$

$$S \Rightarrow SS \Rightarrow aSbS$$

$$S \Rightarrow SS \Rightarrow S$$

$$aabb$$

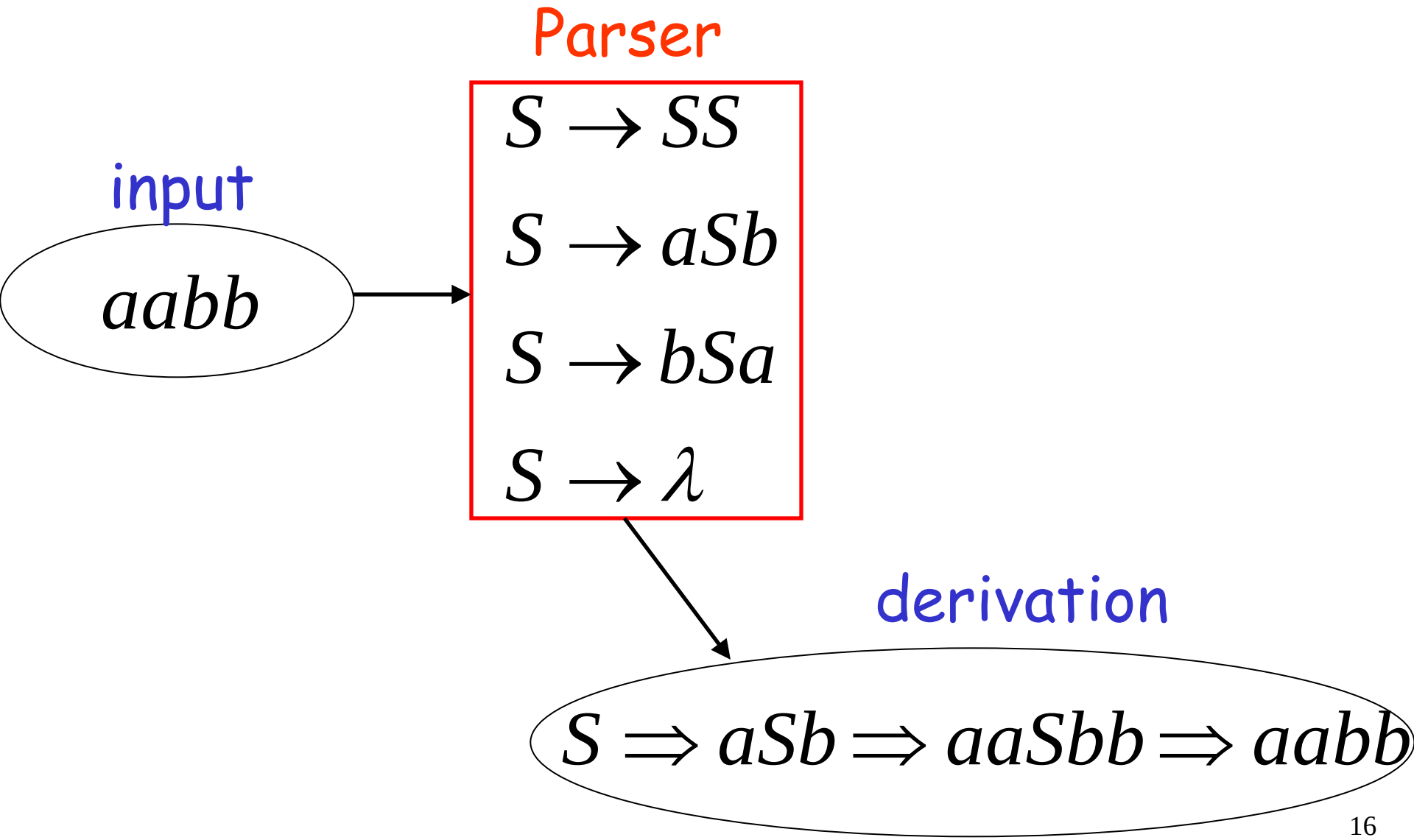
$$S \Rightarrow aSb \Rightarrow aSSb$$

$$S \Rightarrow aSb \Rightarrow aaSbb$$

Phase 3

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$$

Final result of exhaustive search (top-down parsing)



Time complexity of exhaustive search

Suppose there are no productions of the form

$$A \to \lambda$$

$$A \to B$$

Number of phases for string w : $2|w|$

Each step in the derivation increases the length of the sentential form or the number of nonterminal symbols.

Why 2|w| and not just |w|?

Because S=>AS=>ABS=>ABC=>aBC=>abC=>abc

For grammar with k rules

Time for phase 1: k

k possible derivations

Time for phase 2: k^2

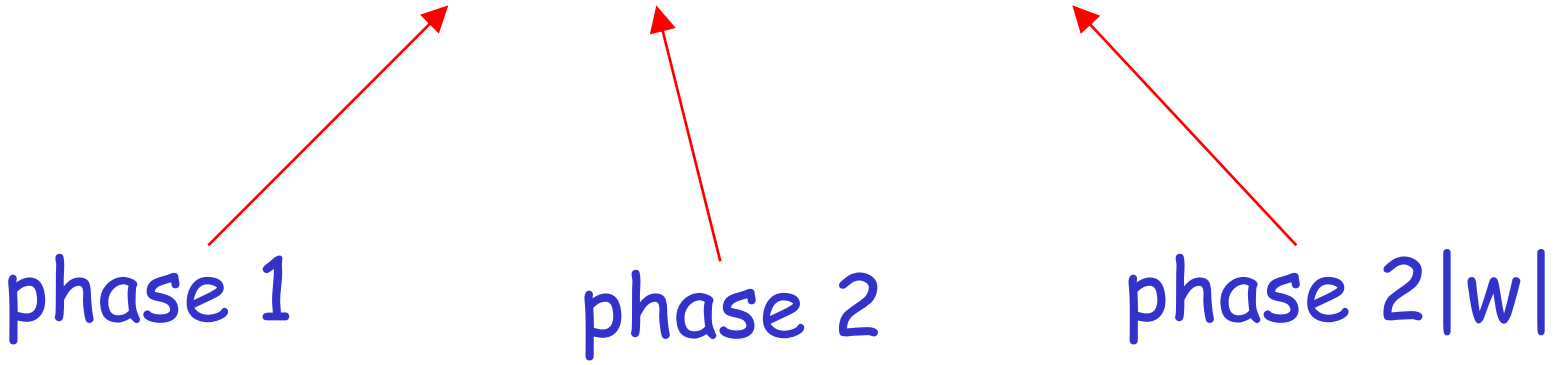
k^2 possible derivations

Time for phase $2|w|$: $k^{2|w|}$

$k^{2|w|}$ possible derivations

Total time needed for string w:

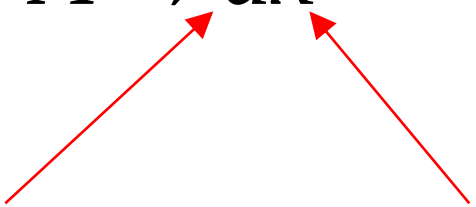
$$k + k^2 + \cdots + k^{2|w|}$$



Extremely bad!!!

There exist faster algorithms for specialized grammars

S-grammar: $A \to ax$



symbol

string of variables

Pair (A, a) appears once

S-grammar example:

$$S \to aS$$

$$S \to bSS$$

$$S \to c$$

Each string has a unique derivation…

$$S \Rightarrow aS \Rightarrow abSS \Rightarrow abcS \Rightarrow abcc$$

For S-grammars:

In the exhaustive search parsing
there is only one choice in each phase

Time for a phase: 1

Total time for parsing string w: $|w|$

For general context-free grammars:

There exists a parsing algorithm that parses a string $|w|$ in time $|w|^3$

Linz 6th, Theorem 5.3, page 144.

Linz 6th, Section 6.3, page 178ff – the CYK algorithm.

Simplifications of Context-Free Grammars

A Substitution Rule

$A \to a$

$A \to aaA$

$A \to abBc$

$B \to abbA$

$B \to b$

Equivalent grammar

$A \to a$

$A \to aaA$

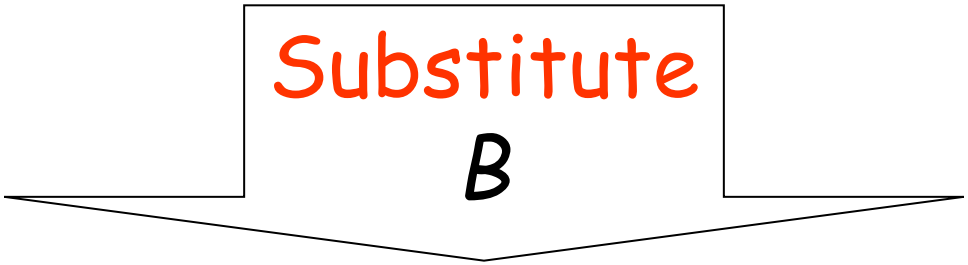
$A \to ababbAc$

$A \to abbc$

In general:

$$A \to xBz$$

$$B \to y_1 \mid y_2 \mid \cdots \mid y_n$$



$$A \to xy_1z \mid xy_2z \mid \cdots \mid xy_nz$$

equivalent grammar

Useless Productions

$$S \to aSb$$

$$S \to \lambda$$

$$S \to A$$

$$A \to aA$$ Useless Production

Some derivations never terminate…

$$S \Rightarrow A \Rightarrow aA \Rightarrow aaA \Rightarrow \ldots \Rightarrow aa\ldots aA \Rightarrow \ldots$$

Another grammar:

$$S \to A$$

$$A \to aA$$

$$A \to \lambda$$

$$B \to bA$$

Useless Production

Not reachable from S

In general:

If $S \Rightarrow \ldots \Rightarrow xAy \Rightarrow \ldots \Rightarrow w$

$w \in L(G)$

Then variable A is useful

Otherwise, variable A is useless

A production $A \to x$ is useful

if all its variables are useful

Removing Useless Productions

Example Grammar:

$$S \to aS \mid A \mid C$$

$$A \to a$$

$$B \to aa$$

$$C \to aCb$$

First: find all variables that produce strings with only terminals

$S \to aS \mid A \mid C$

$A \to a$

$B \to aa$

$C \to aCb$

Round 1: $\{A, B\}$

Round 2: $\{A, B, S\}$

Keep only the variables
that produce terminal symbols

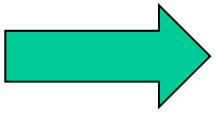
$\{A, B, S\}$

$S \rightarrow aS \mid A \mid C$

$A \rightarrow a$

$B \rightarrow aa$

$C \rightarrow aCb$



$S \rightarrow aS \mid A$

$A \rightarrow a$

$B \rightarrow aa$

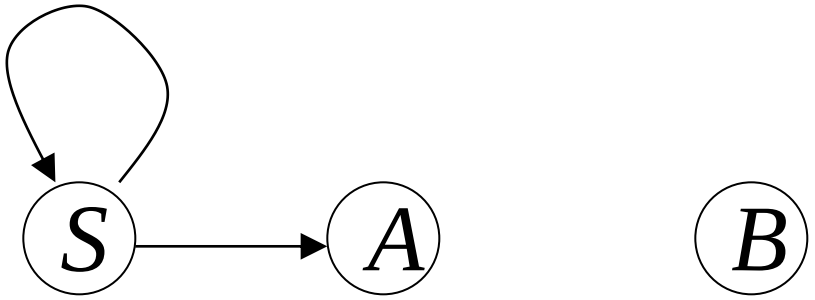
Second: Find all variables reachable from S

$S \to aS \mid A$

$A \to a$

$B \to aa$

Dependency Graph



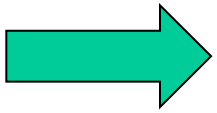
not reachable

Keep only the variables reachable from S

$$S \rightarrow aS \mid A$$

$$A \rightarrow a$$

~~$$B \rightarrow aa$$~~



Final Grammar

$$S \rightarrow aS \mid A$$

$$A \rightarrow a$$

Nullable Variables

λ – production : $A \to \lambda$

Nullable Variable: $A \Rightarrow \ldots \Rightarrow \lambda$

Removing Nullable Variables

Example Grammar:

$$S \rightarrow aMb$$

$$M \rightarrow aMb$$

$$M \rightarrow \lambda$$

Nullable variable

Final Grammar

$S \rightarrow aMb$

$M \rightarrow aMb$

~~$M \rightarrow \lambda$~~

Substitute

$M \rightarrow \lambda$

$S \rightarrow aMb$

$S \rightarrow ab$

$M \rightarrow aMb$

$M \rightarrow ab$

Unit-Productions

Unit Production: $A \to B$

Removing Unit Productions

Observation:

$$A \to A$$

Is removed immediately

Example Grammar:

$$S \to aA$$

$$A \to a$$

$$A \to B$$

$$B \to A$$

$$B \to bb$$

$S \to aA$

$A \to a$

~~$A \to B$~~

$B \to A$

$B \to bb$

Substitute

$A \to B$

$S \to aA \mid aB$

$A \to a$

$B \to A \mid B$

$B \to bb$

$S \rightarrow aA \mid aB$

$A \rightarrow a$

$B \rightarrow A \mid \cancel{B}$

$B \rightarrow bb$

Remove

$B \rightarrow B$

$S \rightarrow aA \mid aB$

$A \rightarrow a$

$B \rightarrow A$

$B \rightarrow bb$

$S \to aA \mid aB$

$A \to a$

~~$B \to A$~~

$B \to bb$

Substitute

$B \to A$

$S \to aA \mid aB \mid aA$

$A \to a$

$B \to bb$

Remove repeated productions

$S \to aA \mid aB \mid aA$

$A \to a$

$B \to bb$

Final grammar

$S \to aA \mid aB$

$A \to a$

$B \to bb$

Removing All

Step 1: Remove Nullable Variables

Step 2: Remove Unit-Productions

Step 3: Remove Useless Variables