Compiler Design

Introduction To Syntax Analysis

Samit Biswas[1]

[1]Department of Computer Science and Technology,
Indian Institute of Engineering Science and Technology, Shibpur
Email: samit@cs.iiests.ac.in

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Table of Contents

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Introduction To Syntax Analysis

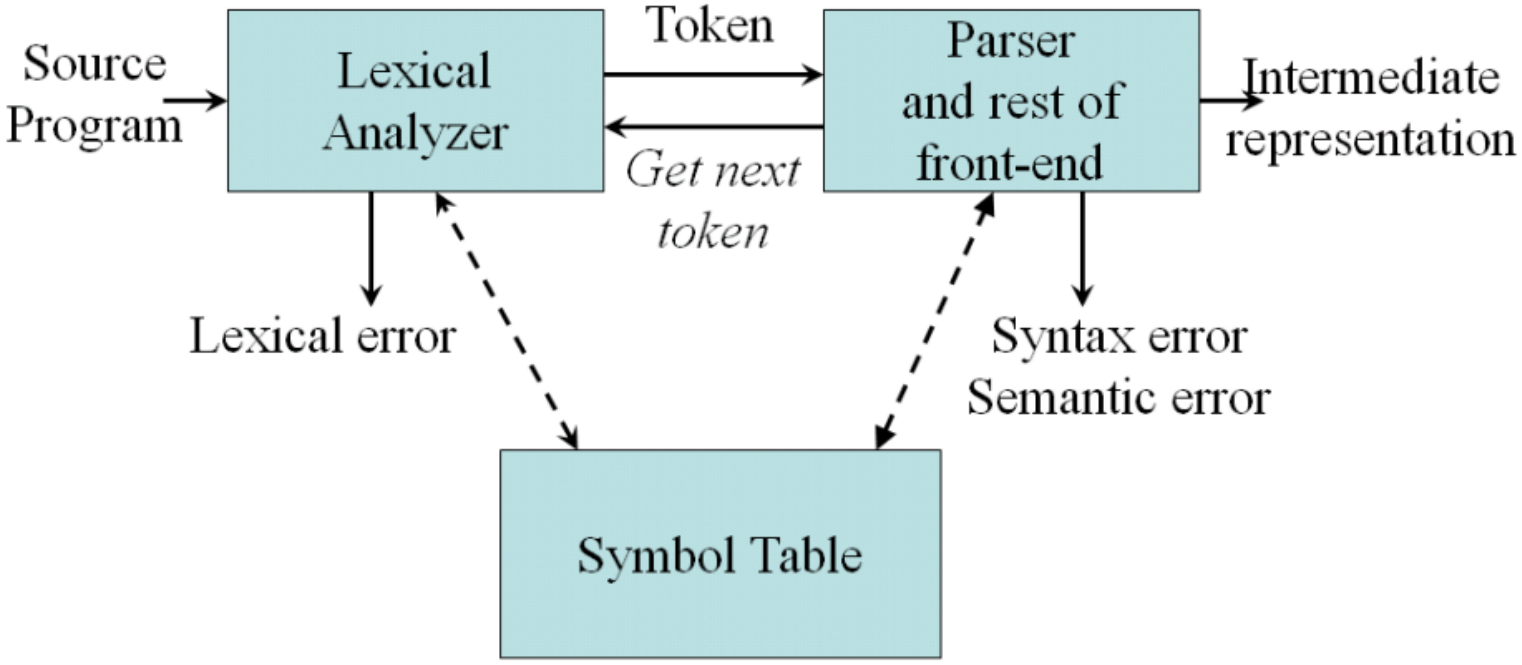
- The syntactic or the structural correctness of a program is checked during the syntax analysis phase of compilation. The structural properties of language constructs can be specified in different ways. Different styles of specification are useful for different purposes.

Introduction To Syntax Analysis

- The syntactic or the structural correctness of a program is checked during the syntax analysis phase of compilation. The structural properties of language constructs can be specified in different ways. Different styles of specification are useful for different purposes.
- The parser (**syntax analyzer**) receives the source code in the form of **tokens** from the lexical analyzer and performs syntax analysis, which create a tree-like intermediate representation that depicts the grammatical structure of the token stream.
  - Syntax analysis is also called parsing.
  - A typical representation is a abstract syntax tree in which
    - each interior node represents an operation
    - the children of the node represent the arguments of the operation

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Syntax Analyzer - Overview



Role of Parser

- Checks the stream of words and their parts of speech (produced by the scanner) for grammatical correctness
- Determines if the input is syntactically well formed
- Guides checking at deeper levels than syntax (static semantics checking)
- Builds an IR representation of the code

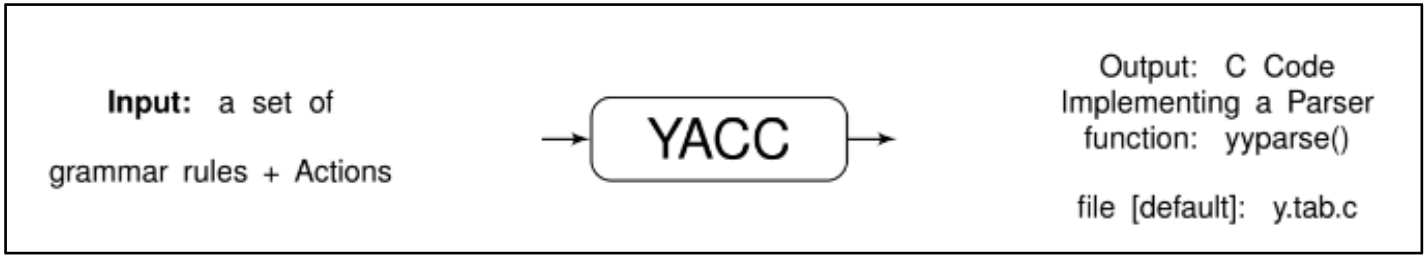
Table of Contents

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

YACC Parser — Overview

Parser generator

- Takes a specification for context-free grammar.
- Produces code for a parser.



Scanner-Parser interaction

- Parser assumes *int yylex()* implements the scanner.

Scanner-Parser interaction

Scanner-Parser interaction

- Parser assumes *int yylex()* implements the scanner.
- Scanner:
  - return value indicates the type of token found;
  - other values communicated using *yytext, yylval* .

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Scanner-Parser interaction

Scanner-Parser interaction

- Parser assumes *int yylex()* implements the scanner.
- Scanner:
  - return value indicates the type of token found;
  - other values communicated using *yytext, yylval* .
- YACC determines integer representations for tokens:
- Communicated to scanner in file `y.tab.h`

  use `yacc -d` to produce `y.tab.h`

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Scanner-Parser interaction

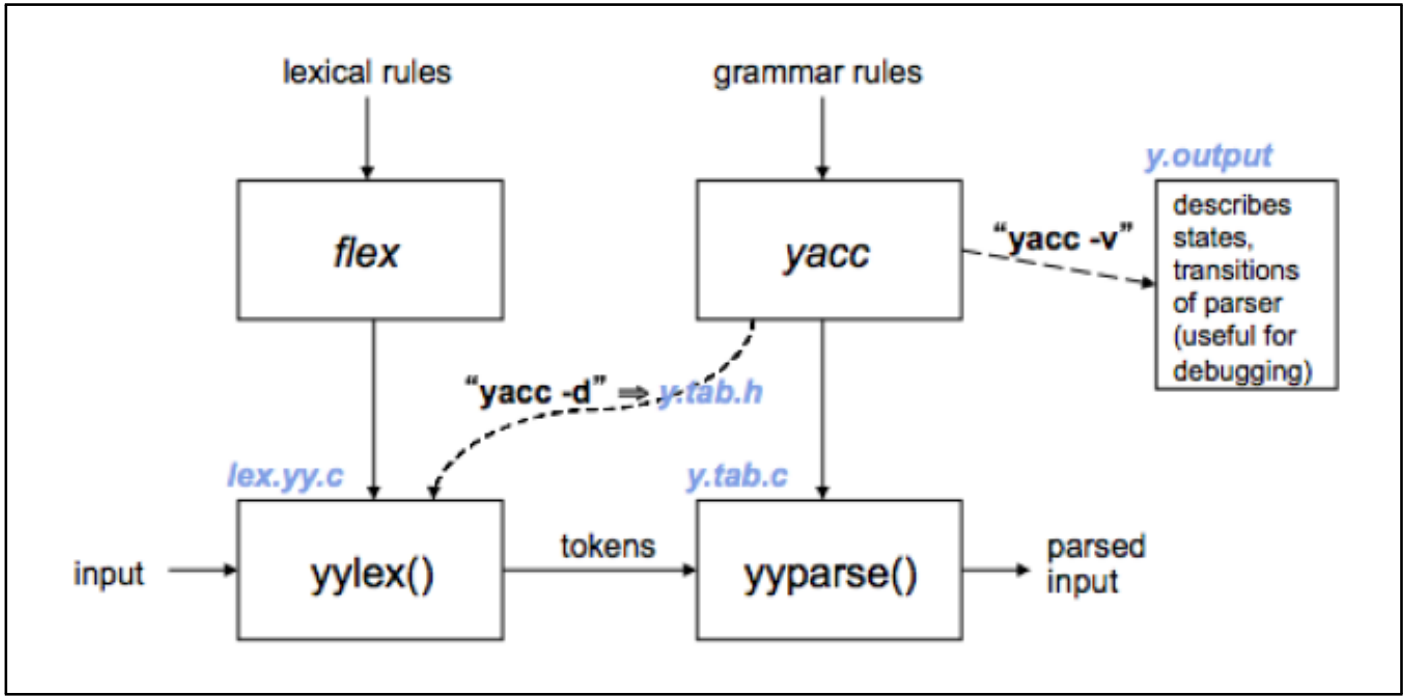
Scanner-Parser interaction

- Parser assumes *int yylex()* implements the scanner.
- Scanner:
  - return value indicates the type of token found;
  - other values communicated using *yytext, yylval* .
- YACC determines integer representations for tokens:
- Communicated to scanner in file `y.tab.h`

  use `yacc -d` to produce `y.tab.h`
- Token encodings:
  - **"end of file"** represented by 0;
  - a character literal: its **ASCII** value;
  - other tokens: assigned numbers $\geq$ **257**

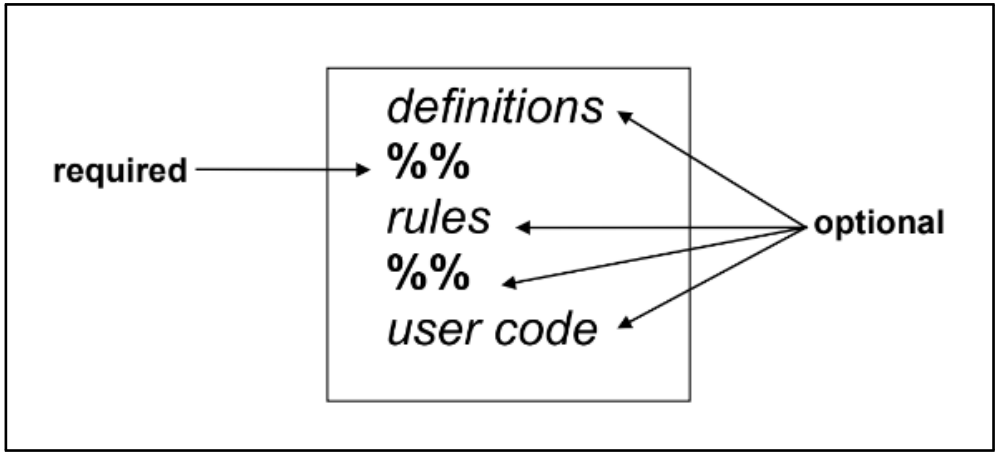
IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Scanner-Parser interaction



flex input format

A YACC input file consists of three parts:



- Shortest possible legal YACC input:

```
%%
```

Definition Section

There are three things that can go in the definitions section:

- C code: Code between % and % is copied to the C file.
- Definitions:
- **Associativity Rules:** These handle associativity and priority of operators.

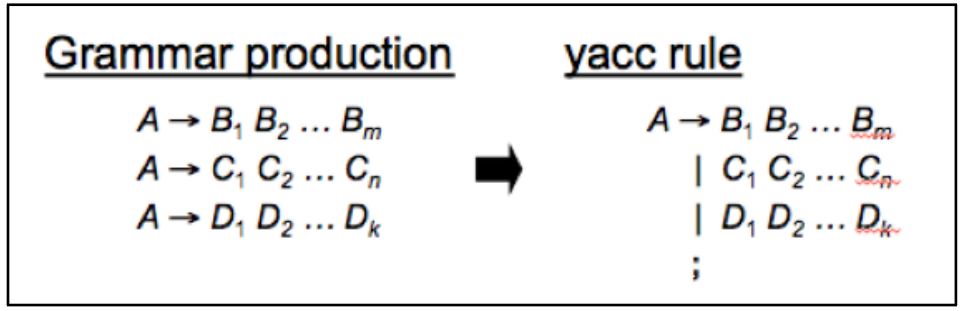
Information about Tokens

- **token names:**
  - declared using **'%token'**.
  - single-character tokens don't have to be declared.
  - any name not declared as a token is assumed to be a nonterminal.
- start symbol of grammar, using **%start [optional]**
- operator info:
  - precedence, associativity.

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Rules section

- The rules section contains the grammar of the language you want to parse. This looks like



- Rule RHS can have arbitrary C code embedded, within . . . .

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

Conflicts

Conflicts

- Conflicts arise when there is more than one way to proceed with parsing.
- Two types:
  - shift-reduce [default action: shift]
  - reduce-reduce [default: reduce with the first rule listed]
- Removing conflicts:
  - specify operator precedence, associativity;
  - restructure the grammar

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

User code section

- The minimal main program is:

```
int main ()
 {
      yyparse();
      return 0;
 }
```

Assignment

Write YACC specification to design a parser for a subset of the C language. Consider the following programming language constructs:

- Main function
- Statements
  - Statements like local/global declarations.
  - Assignment Statements.
  - Conditional Statements.
  - Iterative Statements
  - Function Call
- User defined functions

Consider the assumptions for the language previously considered in last lab sessions. Variables of data types allowed in the subset can be defined at the very beginning of any function / block. A variable must be defined before it is used.

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)

References

- Alfred V. Aho, Ravi Sethi, Jeffrey D Ullman, “Compilers Principles Techniques and Tools”, Pearson Education.

IIEST, Shibpur
आई आई ई एस टि, शिवपुर
Erstwhile B E College (Estd 1856)