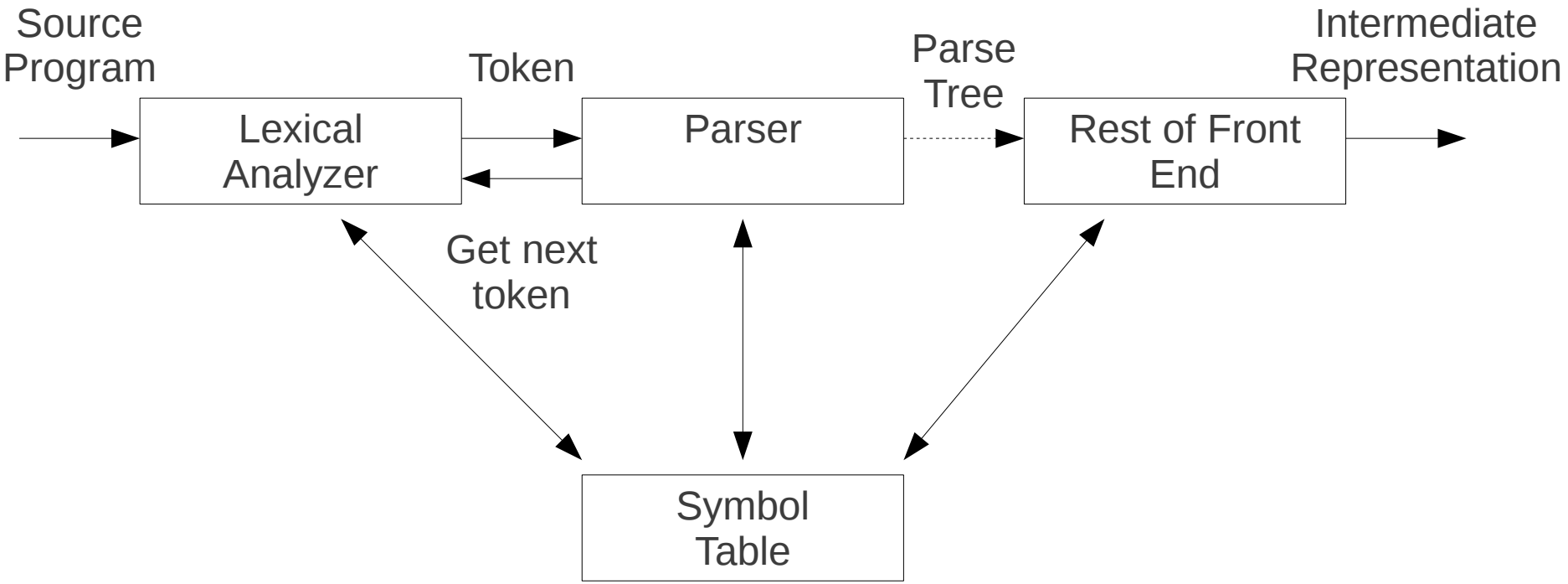
Yacc
(Yet Another Compiler-Compiler)

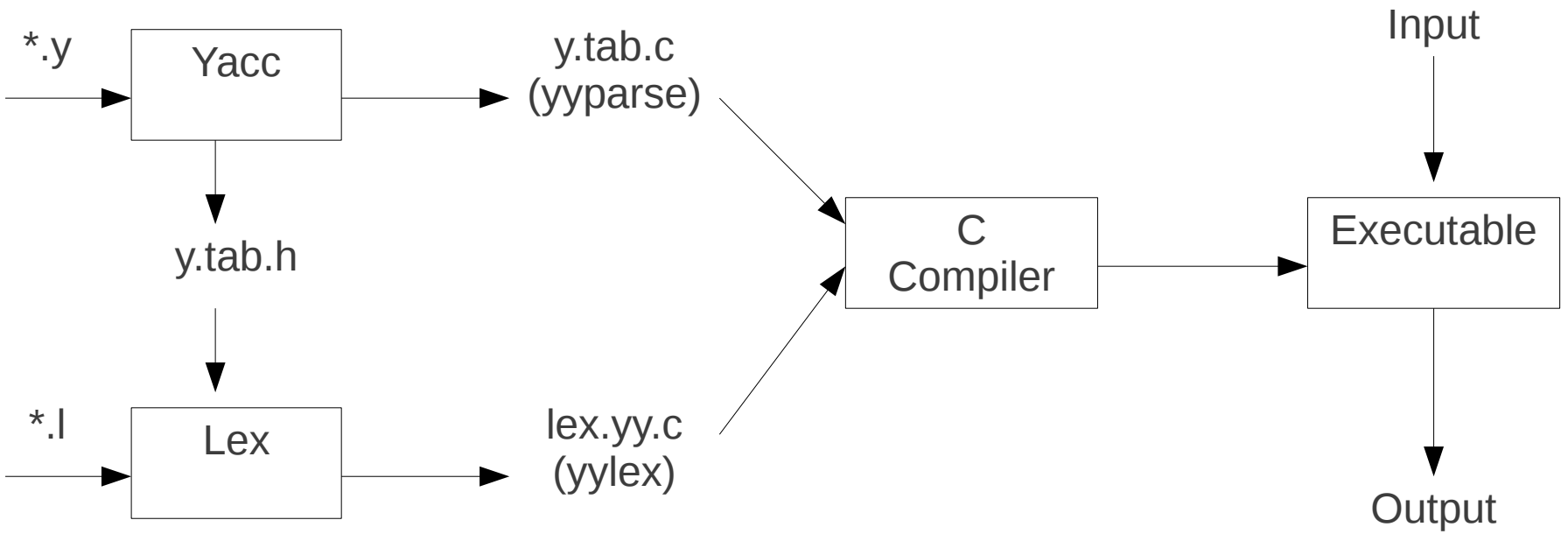
EECS 665 Compiler Construction

Dr. Prasad Kulkarni
Marianne Jantz

The Role of the Parser



Yacc – LALR Parser Generator



Yacc Specification

declarations

%%

translation rules

%%

supporting C routines

Yacc Declarations

- %{ *C declarations* %}
- Declare tokens
  - %token *name1 name2 …*
  - Yacc compiler

    %token INTVAL

    →

    #define INTVAL 257

Yacc Declarations (cont.)

- Precedence
  - %left, %right, %nonassoc
  - Precedence is established by the order the operators are listed (low to high)
- Encoding precedence manually
  - Left recursive = left associative (recommended)
  - Right recursive = right associative

Yacc Declarations (cont.)

- %start
- %union

  %union {

  type *type_name*

  }

  %token <*type_name*> *token_name*

  %type <*type_name*> *nonterminal_name*
- yylval

Yacc Translation Rules

- Form:

  A : Body ;

  where A is a nonterminal and Body is a list of nonterminals and terminals (':' and ';' are Yacc punctuation)
- Semantic actions can be enclosed before or after each grammar symbol in the body
- Ambiguous grammar rules
  - Yacc chooses to shift in a shift/reduce conflict
  - Yacc chooses the first production in reduce/reduce conflict

Yacc Translation Rules (cont.)

- When there is more than one rule with the same left hand side, a '|' can be used

```
A  :  B  C  D  ;

A  :  E  F  ;

A  :  G  ;

=>

A  :  B  C  D
   |  E  F
   |  G
   ;
```

Yacc Actions

- Actions are C code segments enclosed in { } and may be placed before or after any grammar symbol in the right hand side of a rule
- To return a value associated with a rule, the action can set $$
- To access a value associated with a grammar symbol on the right hand side, use $i, where i is the position of that grammar symbol
- The default action for a rule is

  { $$ = $1}

Example of a Yacc Specification

```
%union {
    int ivalue;                     /* Yacc's value stack can hold values of type int */
    char* cvalue;                   /* or character arrays */
}

%token <ivalue> INTVAL              /* define a integer token associated with int value  */
%token <cvalue> ID                  /* define an id token associated with char* value */
%type <ivalue> expr digit           /* define nonterminals associated with int values */
%start program                      /* define starting nonterminal, the overall structure */

%%
program     :  /* empty rule */
            | program ID            { printf( "%s\n", $2 ); }
            | program expr          { printf( "%d\n", $2 ); }

expr        : digit                 { $$ = $1; }
            | expr '+' digit        { $$ = $1 + $3; }
            | expr '-' digit        { $$ = $1 - $3; }

digit       : INTVAL                { $$ = $1; }
%%
```

Corresponding Lex Specification

```
%{
#include <stdlib.h>
#include <stdio.h>
#include <y.tab.h>
%}

ident       [a-zA-Z][a-zA-Z0-9_]
digit       [0-9]
%%

“+”                 { return '+'; }
“-”                 { return '-'; }
{ident}             { yylval.cvalue = strdup( yytext ); return ID; }
{digit}{digit}*     { yylval.ivalue = atoi( yytext ); return INTVAL };

%%
```

Error Handling in Yacc

- Pops symbols until topmost state has an error production, then shifts error onto stack. Then discards input symbols until it finds one that allows parsing to continue. The semantic routine with an error production can just produce a diagnostic message
- yyerror() prints an error message when syntax error is detected

Debugging in Yacc

- -t flag causes compilation of debugging code. To get debug output, the yydebug variable must be set to 1
- -v flag prepares the file *.output. It contains a readable description of the parsing tables and a report on conflicts generated by grammar ambiguities

Yacc References

- Your Compilers Book
- Yacc Tutorials

  http://dinosaur.compilertools.net/yacc/

  http://www.scribd.com/doc/8669780/Lex-yacc-Tutorial

  http://www.cs.man.ac.uk/~pjj/cs212/ho/node7.html

  www.cs.rug.nl/~jjan/vb/yacctut.pdf

  epaperpress.com/lexandyacc/download/LexAndYaccTutorial.pdf

DOT Graph Language References

- Graphviz DOT Language Documentation

  http://www.graphviz.org/Documentation.php
- DOT Language Wikipedia Page

  http://en.wikipedia.org/wiki/DOT_language