COMPILER (CSE 4120)

(Lecture 1: Overview)

Sungwon Jung

Mobile Computing & Data Engineering Lab
Dept. of Computer Science and Engineering
Sogang University
Seoul, Korea
Tel: +82-2-705-8930
Email : jungsung@sogang.ac.kr

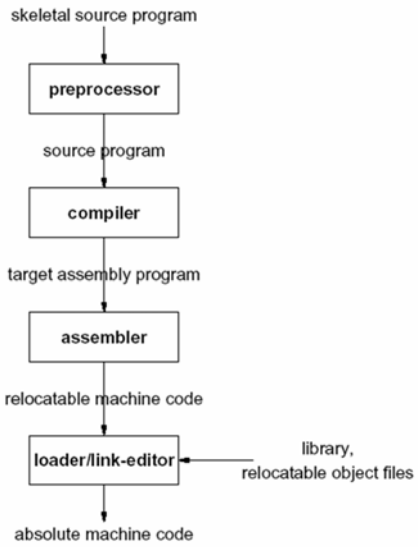
Dept. of Computer Science at Sogang University

What is a compiler ?

- Programming problems are easier to solve in high level languages
  - Language closer to the level of the problem domain
    - FORTRAN : numerical problems
    - Tcl/Tk : graphical user interfaces
- Solutions are usually more efficient (faster, smaller) when written in machine language
  - Language that reflects to the cycle-by-cycle working of a processor
- Compilers are the bridges:
  - Tools to translate program written in high level languages to efficient executable code

The Context of a Compiler

- Language processing system: Typical "compilation"



Dept. of Computer Science at Sogang University

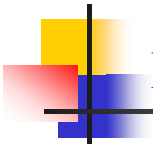
Why study compilers?

- To understand existing language better
- To understand the interface between applications and architectures
- To write compilers for useful domain-specific language (TeX, HTML, Postscript, SQL, …)
- To implement your own language
- To talk intelligently about languages
- To learn a wide range of compiler algorithms and theories
- To learn various tools (Lex, Yacc, debuggers, …)
- To enhance your programming and software engineering skills

Source Code

- Source code: optimized for human readability
  - expressive: matches human grammar
  - redundant

```
int expr(int n)
{
    int d;
    d = 4 * n * n * (n + 1) * (n + 1);
    return d;
}
```

Machine Code

- Optimized for hardware
- Redundancy, ambiguity reduced
- Information about intent lost
- Assembly code: lowest-level source

```
lda $30,-32($30)
stq $26,0($30)
stq $15,8($30)
bis $30,$30,$15
bis $16,$16,$1
stl $1,16($15)
lds $f1,16($15)
sts $f1,24($15)
ldl $5,24($15)
bis $5,$5,$2
s4addq $2,0,$3
ldl $4,16($15)
mull $4,$3,$2
ldl $3,16($15)
addq $3,1,$4
mull $2,$4,$2
ldl $3,16($15)
addq $3,1,$4
mull $2,$4,$2
stl $2,20($15)
ldl $0,20($15)
br $31,$33
$33:
bis $15,$15,$30
ldq $26,0($30)
ldq $15,8($30)
addq $30,32,$30
ret $31,($26),1
```

What do you generally expect from a compiler?

- Correct compilation
- Good performance of the target program
- Short compilation time
- Detailed and specific error reports

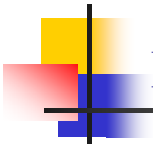
Example (Output assembly code)

- Unoptimized Code

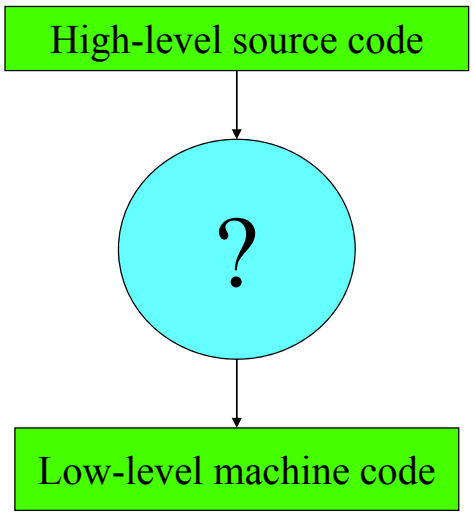
```
    lda $30,-32($30)
    stq $26,0($30)
    stq $15,8($30)
    bis $30,$30,$15
    bis $16,$16,$1
    stl $1,16($15)
    lds $f1,16($15)
    sts $f1,24($15)
    ldl $5,24($15)
    bis $5,$5,$2
    s4addq $2,0,$3
    ldl $4,16($15)
    mull $4,$3,$2
    ldl $3,16($15)
    addq $3,1,$4
    mull $2,$4,$2
    ldl $3,16($15)
    addq $3,1,$4
    mull $2,$4,$2
    stl $2,20($15)
    ldl $0,20($15)
    br $31,$33
$33:
    bis $15,$15,$30
    ldq $26,0($30)
    ldq $15,8($30)
    addq $30,32,$30
    ret $31,($26),1
```

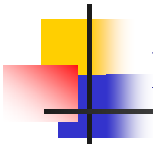
Optimized Code

```
s4addq $16,0,$0
mull $16,$0,$0
addq $16,1,$16
mull $0,$16,$0
mull $0,$16,$0
ret $31,($26),1
```



How to translate effectively?

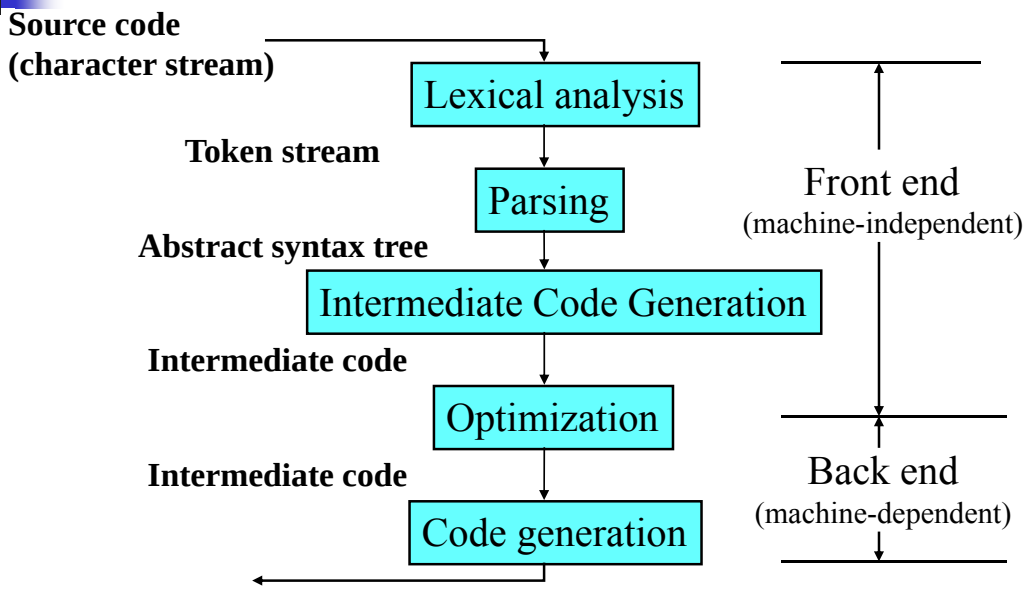




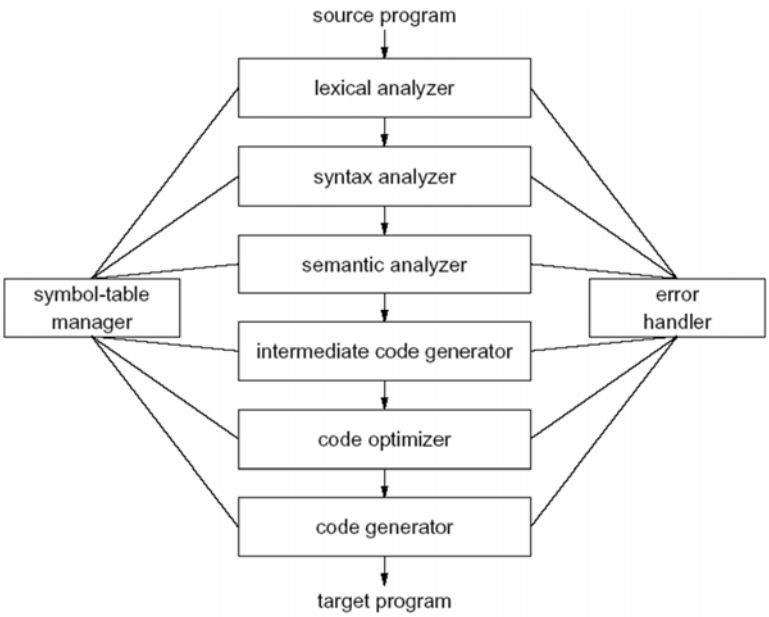
Idea: Translate in Steps

- Series of program representations
- Intermediate representations optimized for program manipulations of various kinds (checking and optimization)
- Become more machine-specific, less language-specific as translation proceeds

Standard Compiler Structure



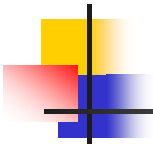
The phase of a Compiler



Role of Phases

1. Lexical analyzer (scanner):
   - input characters → tokens
2. Syntax analyzer (parser):
   - tokens → syntax tree
3. Semantic analyzer:
   - type checking, etc.
4. Intermediate code generator:
   - syntax tree → intermediate code
5. Code optimizer:
   - improvements to intermediate code
6. Final code generator:
   - intermediate code → executable/assembly code

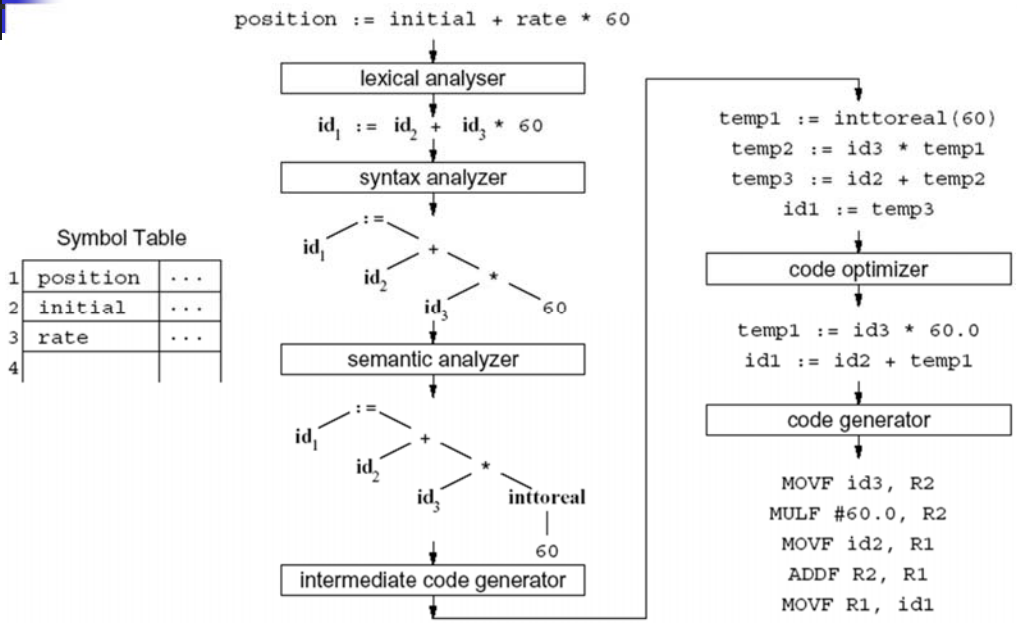
Dept. of Computer Science at Sogang University



Symbol Table Management

- Symbol Table
  - Data structure containing a record for each identifier with attributes
  - Attribute: storage allocation information – identier, type, scope, etc.
  - Collection of records organized into lists, arrays or hash tables

Translation of a Statement



First Step: Lexical Analysis