Lecture 06
Semantic Analysis

Outline

- Semantic Analysis
  - Overview of Semantic Analysis
  - Attributes and Attribute Grammars
  - Dependency Graphs and Algorithms for Attribute Computation
  - Symbol Table and Scope Checking
  - Type Checking for Semantic Analysis of a Program

V. Semantic Analysis of a Program

1. Declarations

- Typically, the information in declarations is inserted into a symbol table for later lookup during the translation of other parts of the program
- Assume that insert(id.name,dtype) is a procedure that insets an identifier into the symbol table and associates a type to it
- Attribute grammar is as follow:

Attribute grammar is as follow:

Grammar rule	Semantic Rules
decl->type varlist	varlist.dtype=type.dtype
type->int	type.dtype=integer
type->float	type.dtype=real
varlist1->id,varlist2	insert(id.name,varlist1.dtype) varlist2.dtype =varlist1.dtype
var-list->id	insert(id.name,varlist.dtype)

2. Statements

- Semantic analysis of statements is mainly type checking (the use of type information to ensure that each part of a program makes sense under the type rules of the language)
- A simple grammar to illustrate semantic analysis

```
stmt->id:=exp
stmt->if exp then stmt
exp-> exp1+exp2
exp->exp1 or exp2
exp-> id
```

Attributes and Procedures used in attribute grammar

- We assume the availability of a symbol table that contains variable names and associated data types

Attribute:

- name of an identifier
- dtype of grammar symbol

Procedures:

- lookup(id.name), which returns the associated type of a name if it has already in the symbol table, otherwise returns nil
- error, which reports semantic errors

Attribute grammar for semantic analysis (Type Checking) of simple grammar

Grammar rule	Semantic Rules
exp→exp1+exp2	if exp1.type≠integer or exp2.type≠integer then error else exp.type=integer
exp→exp1 or exp2	if exp1.type ≠boolean or exp2.type ≠boolean then error else exp.type=boolean

exp→id	t=lookup(id.name)
	if t ≠nil then exp.type=t else error

id→num	id.type = integer
stmt→id:=exp	t=lookup(id.name) if t=nil then error else if t ≠ exp.type then error
stmt→if exp then stmt	if exp.type ≠ boolean then error

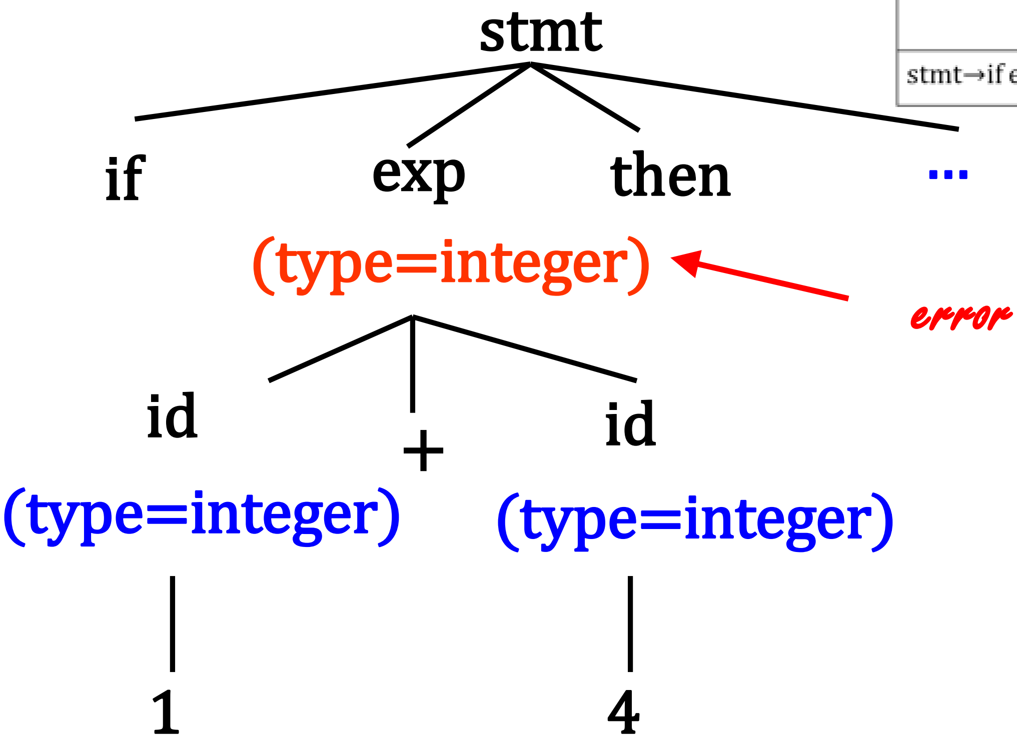
Type Checking(An example)

Grammar rule	Semantic Rules
exp→exp1+exp2	if exp1.type≠integer or exp2.type≠integer then error else exp.type=integer
exp→exp1 or exp2	if exp1.type ≠boolean or exp2.type ≠boolean then error else exp.type=boolean

exp→id	t=lookup(id.name)
	if t ≠nil then exp.type=t else error

id→num	id.type = integer
stmt→id:=exp	t=lookup(id.name) if t=nil then error else if t ≠ exp.type then error
stmt→if exp then stmt	if exp.type ≠ boolean then error

if(1 + 4) then /*   …   */


stmt
if
exp
then
...
(type=integer)
error
id
+
id
(type=integer)
(type=integer)
1
4