编译原理第六章第二次作业

李昊宸 2017K8009929044

6.3.1 确定下列声明序列中各个标识符的类型和相对地址。

float x;

record {float x; float y;} p;

record {int tag; float x; float y;} q;

答:

id	type	offset
x	float	0
x	float	0
y	float	8
p	record	8
tag	int	0
x	float	4
y	float	12
q	record	24

6.4.3 使用图 6-22 的翻译方案来翻译下列赋值语句:

1) x = a[i] + b[j]

2) x = a[i][j] + b[i][j]

$S \rightarrow \mathbf{id} = E\ ;$ { $gen(top.get(\mathbf{id}.lexeme)\ '='\ E.addr);$ }

| $L = E\ ;$ { $gen(L.array.base\ '['\ L.addr\ ']'\ '='\ E.addr);$ }

$E \rightarrow E_1 + E_2$ { $E.addr = \mathbf{new}\ Temp();$
$gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr);$ }

| $\mathbf{id}$ { $E.addr = top.get(\mathbf{id}.lexeme);$ }

| L { $E.addr = \mathbf{new}\ Temp();$
$gen(E.addr\ '='\ L.array.base\ '['\ L.addr\ ']');$ }

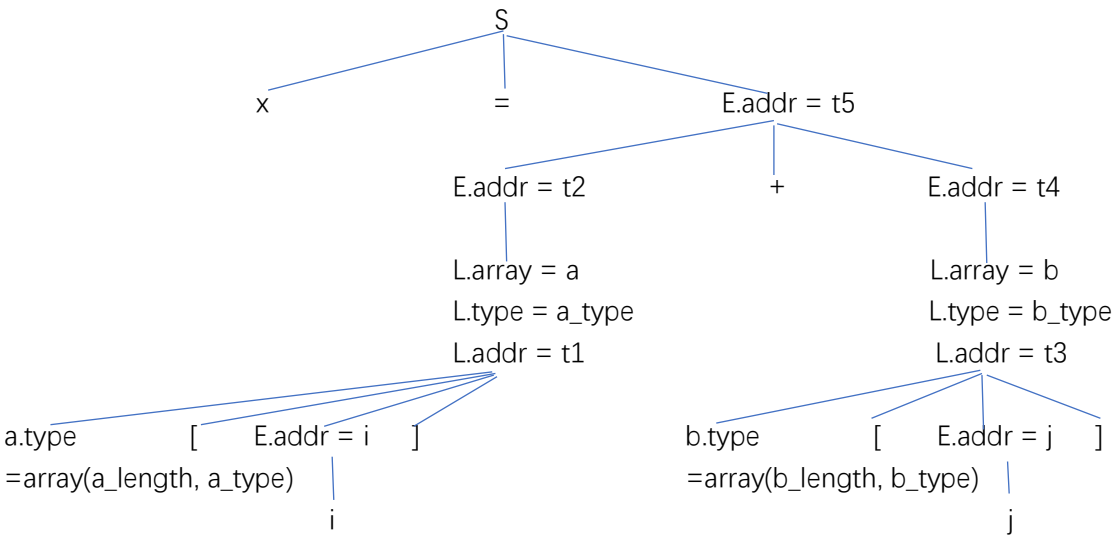
$L \rightarrow \mathbf{id}\ [\ E\ ]$ { $L.array = top.get(\mathbf{id}.lexeme);$
$L.type = L.array.type.elem;$
$L.addr = \mathbf{new}\ Temp();$
$gen(L.addr\ '='\ E.addr\ '*'\ L.type.width);$ }

| $L_1\ [\ E\ ]$ { $L.array = L_1.array;$
$L.type = L_1.type.elem;$
$t = \mathbf{new}\ Temp();$
$L.addr = \mathbf{new}\ Temp();$
$gen(t\ '='\ E.addr\ '*'\ L.type.width);$
$gen(L.addr\ '='\ L_1.addr\ '+'\ t);$ }

图 6-22 处理数组引用的语义动作

答:

1） x = a[i] + b[j]



t1 = i * a_type.width

t2 = a [t1]

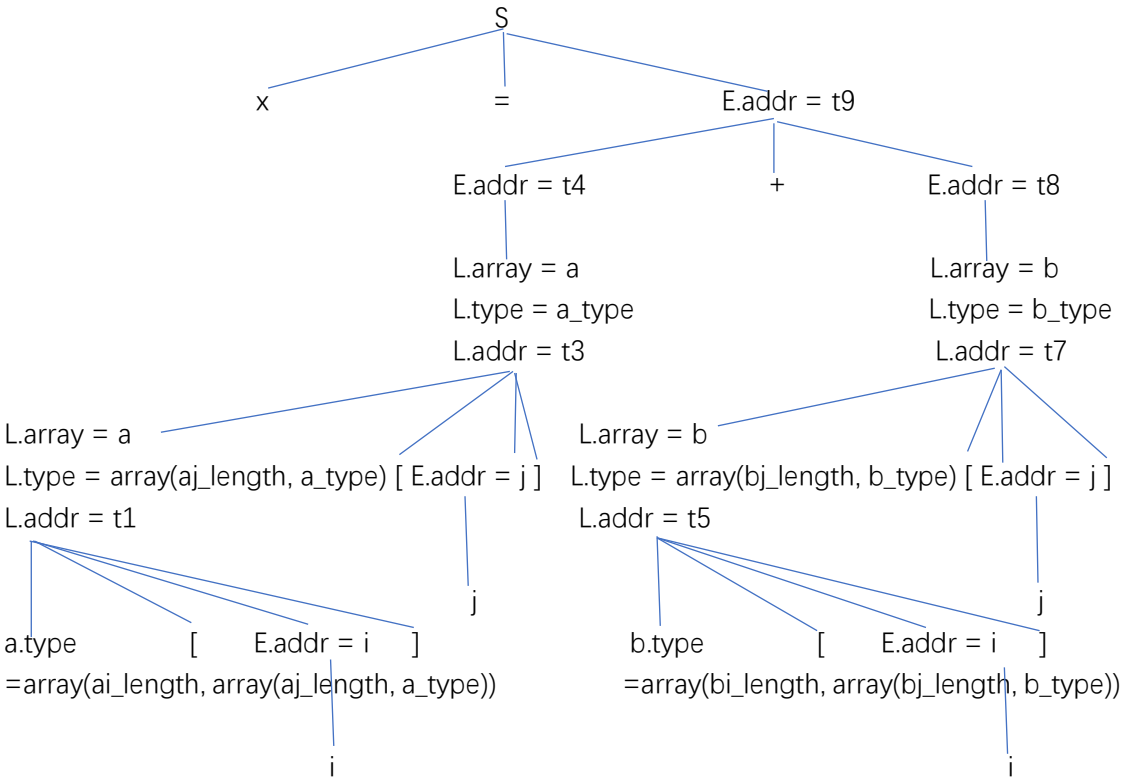
t3 = j * b_type.width

t4 = b [t3]

t5 = t2 + t4

x = t5

2) x = a[i][j] + b[i][j]

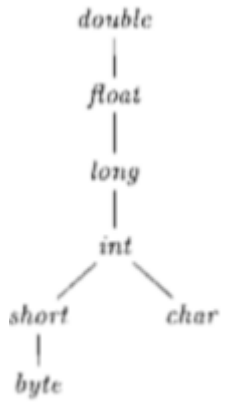


t1 = i * array(aj_length, a_type).width
t2 = j * a_type
t3 = t1 + t2
t4 = a [t3]
t5 = i * array(bj_length, b_type).width
t6 = j * b_type
t7 = t5 + t6
t8 = b [t7]
t9 = t4 + t8
x = t9

6.5.1 假定图 6-26 中的函数 widen 可以处理图 6-25a 的层次结构中的所有类型，翻译下列表达式。假定 c 和 d 是字符类型，s 和 t 是短整型， i 和 j 为整型， x 是浮点型。
1）x = s + c
2）i = s + c
3）x = (s + c) * (t + d)

```
Addr widen(Addr a, Type t, Type w)
        if ( t = w ) return a;
        else if ( t = integer and w = float ) {
                temp = new Temp();
                gen(temp '=' '(float)' a);
                return temp;
        }
        else error;
}
```

图 6-26 *widen* 函数的伪代码



a) 拓宽类型转换

答：widen 函数原本只能将 integer 转换为 float，现假设该函数可以在所有类型间自下而上的转换

1）x = s + c
t1 = (int) s
t2 = (int) c
t3 = t1 + t2
x = (float) t3

2) i = s + c
t1 = (int) s
t2 = (int) c
i = t1 + t2

3) x = (s + c) * (t + d)
t1 = (int) s

```
t2 = ( int ) c
t3 = t1 + t2
t4 = ( int ) t
t5 = ( int ) d
t6 = t4 + t5
t7 = t3 * t6
x = ( float ) t7
```