中间代码生成
Part5：类型表达式及自动构造

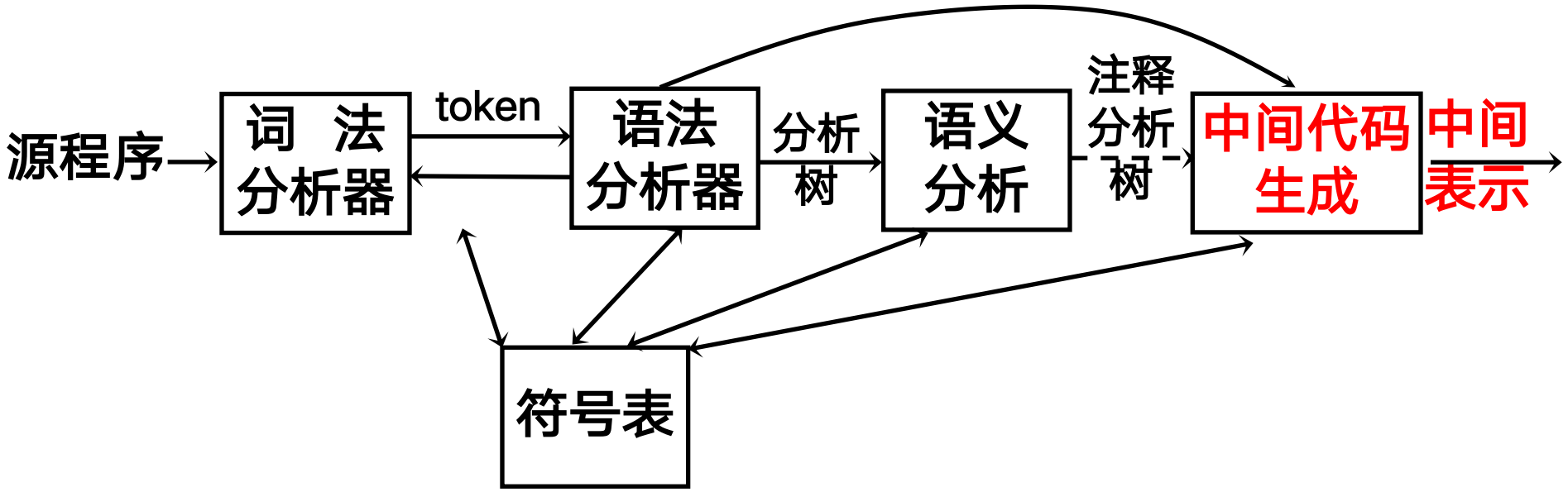
李 诚

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年04月03日

本节提纲



- 类型表达式
- 构造类型表达式的语法制导定义SDD
- 构造类型表达式的语法制导翻译SDT

类型表达式 (Type expression)

- 类型可以是语法的一部分，因此也是结构的

考虑以下文法，D代表声明语句，S代表一般语句

$P \rightarrow D\ ;\ S$

$D \rightarrow D\ ;\ D \mid \mathrm{id}\ :\ T$

$T \rightarrow \mathrm{boolean} \mid \mathrm{integer} \mid \mathrm{array}\ [\mathrm{num}\ ]\ \mathrm{of}\ T \mid \uparrow T \mid T\ \text{‘}\rightarrow\text{’}\ T$

类型表达式 (Type expression)

- 类型可以是语法的一部分，因此也是结构的

考虑以下文法，D代表声明语句，S代表一般语句

$P \rightarrow D\ ;\ S$

$D \rightarrow D\ ;\ D \mid \mathrm{id}\ :\ T$

$T \rightarrow \mathrm{boolean} \mid \mathrm{integer} \mid \mathrm{array}\ [\mathrm{num}\ ]\ \mathrm{of}\ T \mid \uparrow T \mid T\ \text{‘}\rightarrow\text{’}\ T$

数组　指针　函数

基本类型　　复杂且可组合的类型

类型表达式 (Type expression)

- **基本类型是类型表达式**
  - *integer*
  - *real*
  - *char*
  - *boolean*
  - *type_error* //出错类型
  - *void* //无类型

在类型检查中传递错误

语句的类型

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
    - 如果T是类型表达式，N是一个整数，则*array(N, T)*是类型表达式

类型表达式 (Type expression)

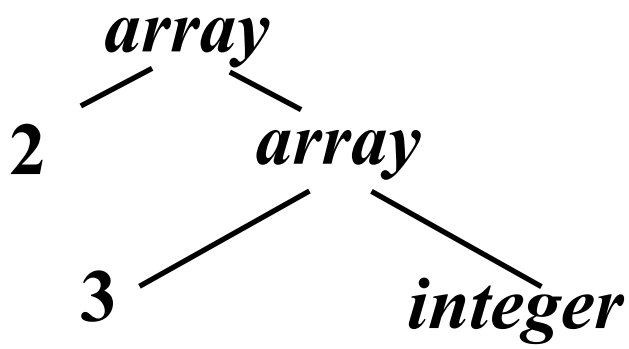
- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
    - 如果T是类型表达式，N是一个整数，则*array(N, T)*是类型表达式

类型	类型表达式
int[3]	*array*（3，*integer*）
int[2][3]	*array*（2，*array*（3，*integer*））

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
    - 如果T是类型表达式，N是一个整数，则*array(N, T)*是类型表达式

类型	类型表达式
int[3]	*array*（3，*integer*）
int[2][3]	*array*（2，*array*（3，*integer*））

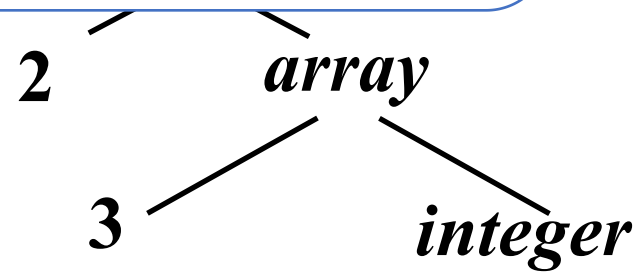


类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
    - 如果T是类型表达式，N是一个整数，则*arr*

类型	类型表达式
int[3]	*array*（3，*integer*）
int[2][3]	*array*（2，*array*（3，*integer*））

也可以写为
array ({0,…,2}, integer)
其中{0,…,2}代表索引集合
如首元素索引从1开始，则
为*array ({1,…,3}, integer)*



类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
  - 指针类型构造算子*pointer*
    - 如果T是类型表达式，则*pointer(T)*是类型表达式

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
  - 指针类型构造算子*pointer*
  - 笛卡尔乘积类型构造算子×
    - 如果T_1和T_2是类型表达式，则$T_1 \times T_2$也是类型表达式
    - 主要用于描述列表和元组，如：表示函数的参数

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
  - 指针类型构造算子*pointer*
  - 笛卡尔乘积类型构造算子×
  - 函数类型构造算子→
    - 若$T_1, T_2, \ldots, T_n$和R是类型表达式，则$T_1 \times T_2 \times \ldots \times T_n \rightarrow R$也是

函数参数　函数返回值

类型表达式 (Type expression)

- 基本类型是类型表达式
- 可为类型表达式命名，类名也是类型表达式
- 将类型构造算子(type constructor)作用于类型表达式可以构成新的类型表达式
  - 数组类型构造算子*array*
  - 指针类型构造算子*pointer*
  - 笛卡尔乘积类型构造算子×
  - 函数类型构造算子→
  - 记录类型构造算子*record*
    - 若有标识符$N_1, N_2, \ldots, N_n$以及对应的类型表达式$T_1, T_2, \ldots, T_n$，则*record*$((N_1\times T_1)\times(N_2\times T_2)\times\ldots\times(N_n\times T_n))$也是类型表达式

记录中的字段

字段对应的类型表达式

类型表达式-例1

- 考虑C语言中数组double a[10][20]，写出a、a[0]、a[0][0]的类型表达式

类型表达式-例1

- **考虑C语言中数组double a[10][20]，写出a、a[0]、a[0][0]的类型表达式**

a[0][0]:　double

类型表达式-例1

- **考虑C语言中数组double a[10][20]，写出a、a[0]、a[0][0]的类型表达式**

a[0][0]: double

a[0]: array(20,double);
pointer(double)

类型表达式-例1

- **考虑C语言中数组double a[10][20]，写出a、a[0]、a[0][0]的类型表达式**

a[0][0]: double

a[0]: array(20,double);
pointer(double)

a: array(10,array(20,double));
pointer(array(20,double))

类型表达式-例2

- **为row、table和p分别写出类型表达式：**

```
typedef struct{
    int address;
    char lexeme[15];
} row;
row table[101];
row *p;
```

类型表达式-例2

- **为row、table和p分别写出类型表达式：**

```
typedef struct{
    int address;
    char lexeme[15];
} row;
row table[101];
row *p;
```

row的类型表达式：

record((address×integer) × (lexeme × (array(15, char))))

table的类型表达式：

array(101, row) //此处row是类型名，因此也是类型表达式

p的类型表达式：

pointer(row)

类型表达式-例3

- **考虑下面的函数f，写出其类型表达式。**

  int *f(char a, char b);

  f的类型表达式:

  (char×char) → pointer(integer)

类型表达式-例3

- **考虑下面的函数f，写出其类型表达式。**

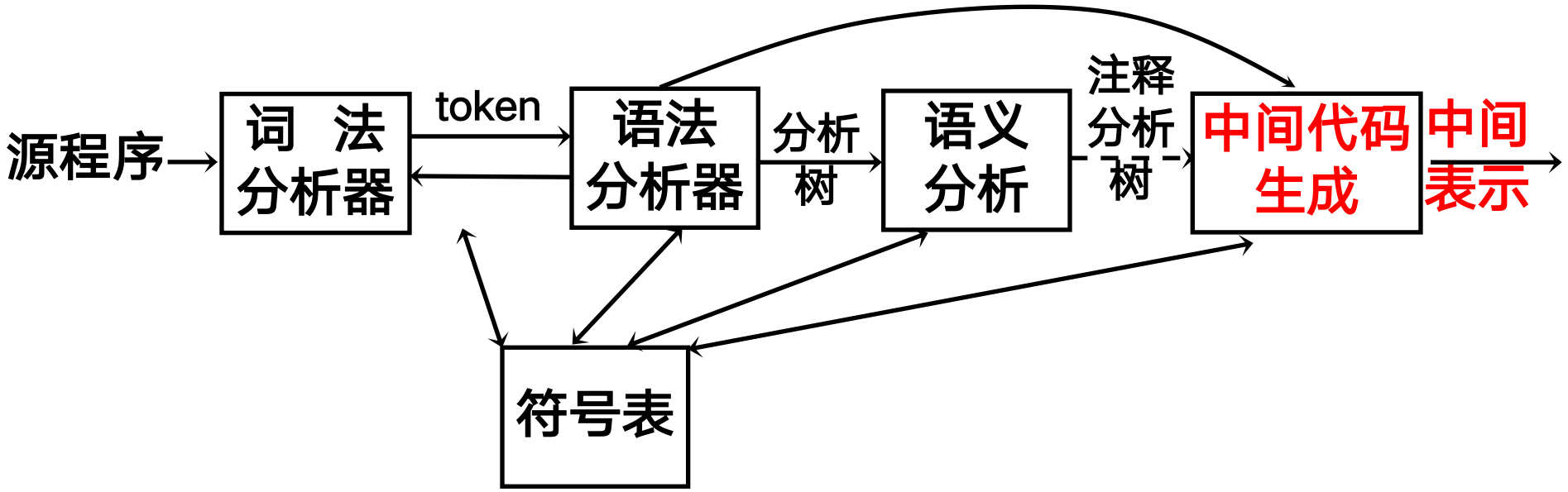
  int *f(char a, char b);

  f的类型表达式:

  (char×char) → pointer(integer)

问题：可否自动化地实现类型表达式的生成？

本节提纲



- 类型表达式
- **构造类型表达式的语法制导定义SDD**
- **构造类型表达式的语法制导翻译SDT**

构造类型表达式的SDD

- 为以下文法制定构造类型表达式的语义规则

产 生 式	语 义 规 则
$T \rightarrow B\ C$	
$B \rightarrow$ int	
$B \rightarrow$ float	
$C \rightarrow$ [num] C_1	
$C \rightarrow \varepsilon$	

构造类型表达式的SDD

- 为以下文法制定构造类型表达式的语义规则

产 生 式	语 义 规 则
$T \rightarrow B\ C$	
$B \rightarrow$ int	
$B \rightarrow$ float	
$C \rightarrow$ [num] C_1	
$C \rightarrow \varepsilon$	

- 为每个文法符号设置综合属性t和继承属性b
  - t: 该符号对应的类型表达式
  - b: 将类型信息从左到右传递

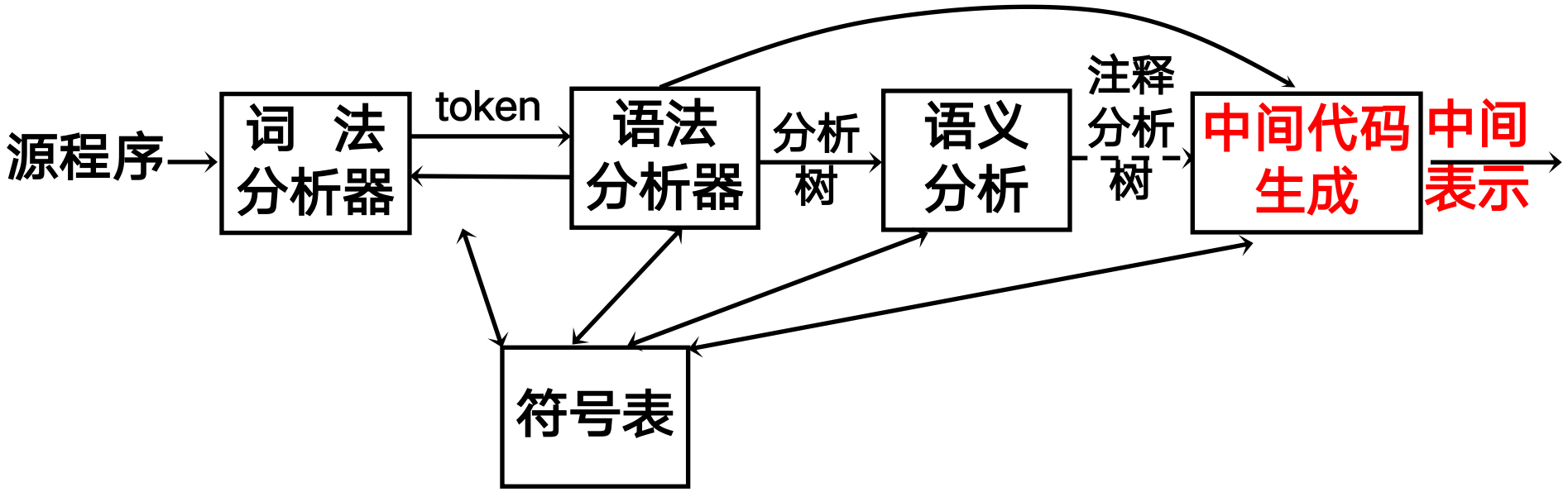
构造类型表达式的SDD

- 为以下文法制定构造类型表达式的语义规则

产 生 式	语 义 规 则
$T \rightarrow B\ C$	$T.t = C.t;\ C.b = B.t$
$B \rightarrow$ int	$B.t = integer$
$B \rightarrow$ float	$B.t = float$
$C \rightarrow$ [num] C_1	$C.t = array(num.val, C_1.t);\ C_1.b = C.b$
$C \rightarrow \varepsilon$	$C.t = C.b$

- 为每个文法符号设置综合属性t和继承属性b
  - t: 该符号对应的类型表达式
  - b: 将类型信息从左到右传递

本节提纲



- 类型表达式
- 构造类型表达式的语法制导定义SDD
- **构造类型表达式的语法制导翻译SDT**

构造类型表达式的SDT

- 将SDD改造为SDT

$T \rightarrow B\ \{C.b = B.t\}\ C\ \{T.t = C.t;\ \}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ \{C_1.b = C.b\}\ C_1\ \{C.t = array(num.val,\ C_1.t);\ \}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$

- 但是继承属性的计算与LR分析方法不适配
- 因此，如果要使用LR，就需要改造文法

构造类型表达式的SDT

- **通过改造文法，与LR适配**
  - 引入标记M，C归约时可在栈顶以下位置找到B.t
  - 引入标记N，把继承属性C.b当做综合属性记录

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ N\ C_1\ \{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$

构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

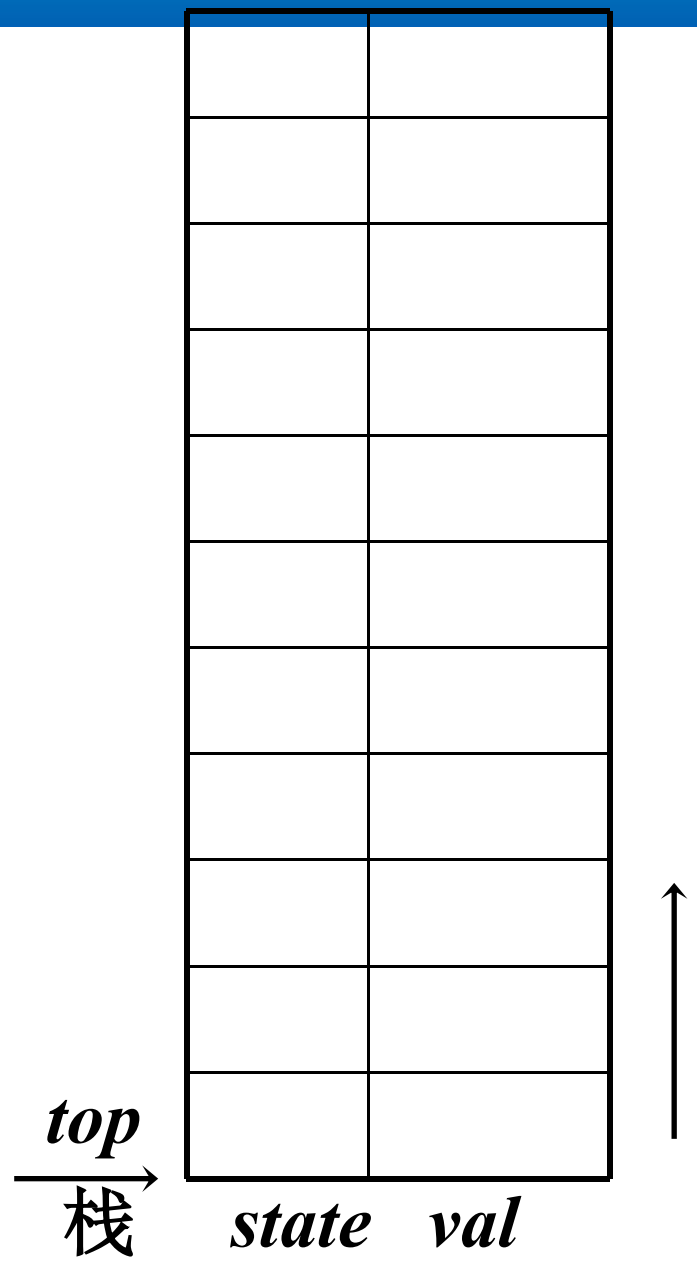
$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ N\ C_1$

$\{C.t = array(num.val,\ C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

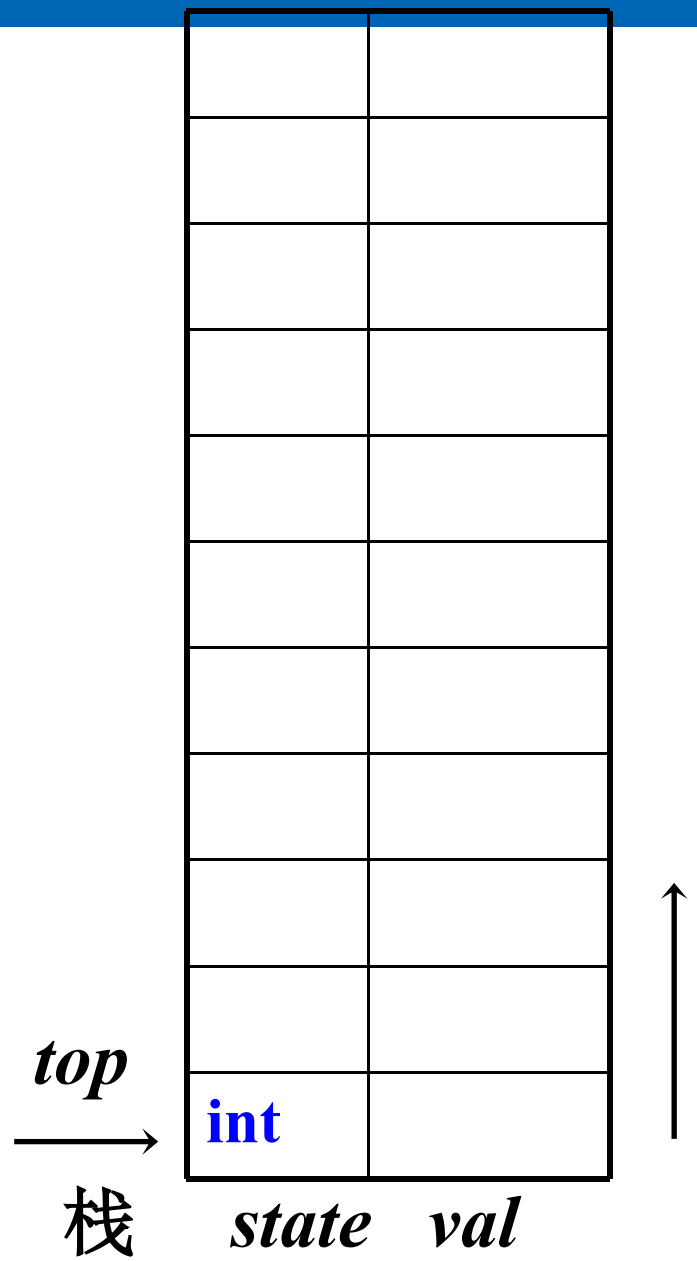
$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ N\ C_1$

$\{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

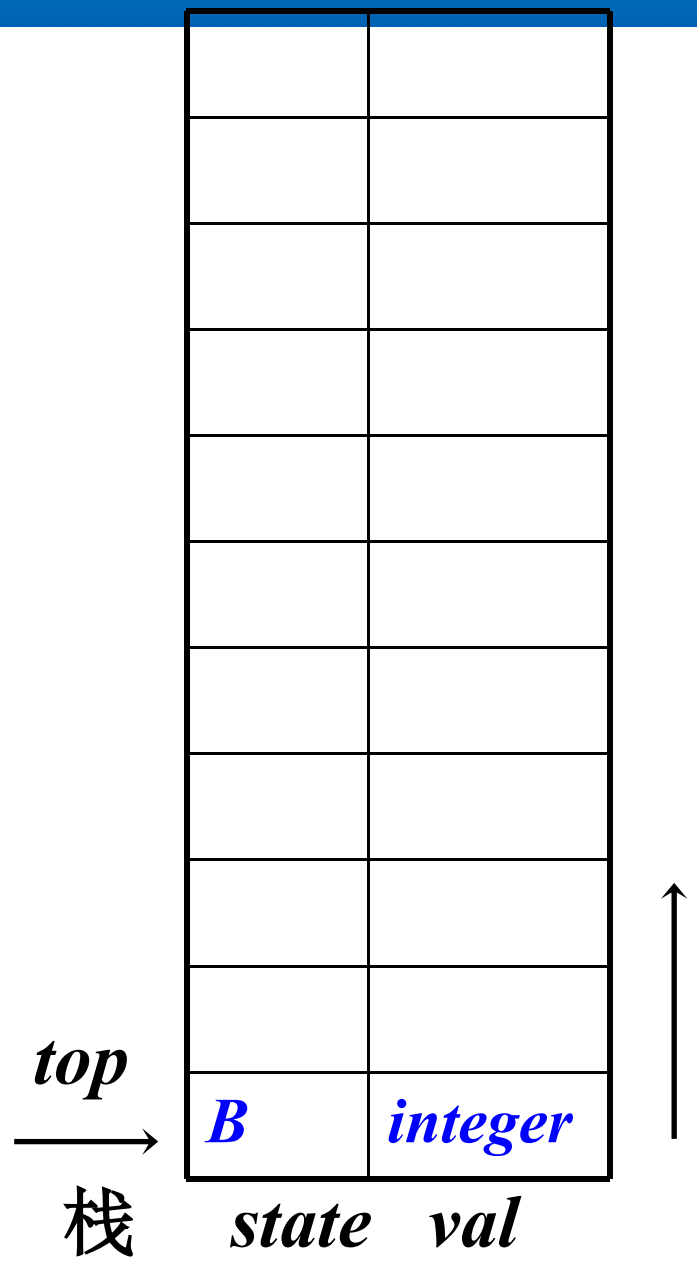
$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ N\ C_1$

$\{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow \text{int}\ \{B.t = integer\}$

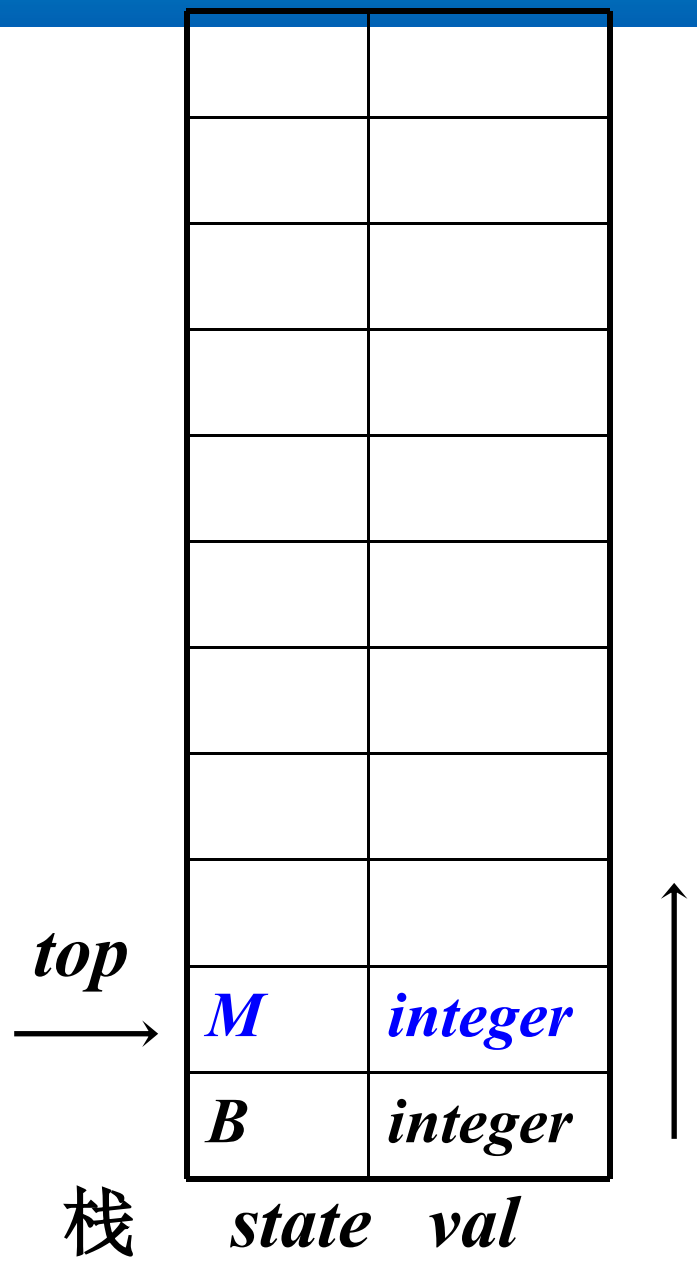
$B \rightarrow \text{float}\ \{B.t = float\}$

$C \rightarrow [\text{num}]\ N\ C_1$

$\{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow$ int $\{B.t = integer\}$

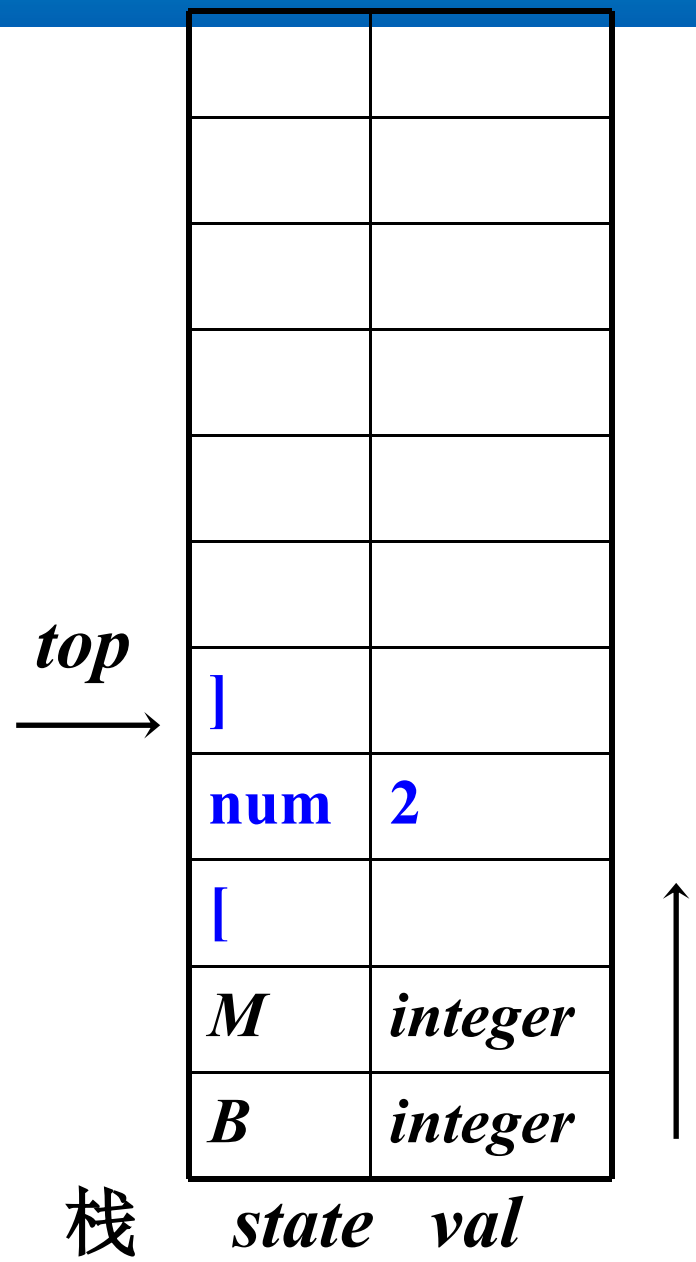
$B \rightarrow$ float $\{B.t = float\}$

$C \rightarrow$ [num] $N\ C_1$

$\{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C\ \{T.t = C.t;\ \}$

$M \rightarrow \varepsilon\ \{M.t = B.t\}$

$B \rightarrow$ int $\{B.t = integer\}$

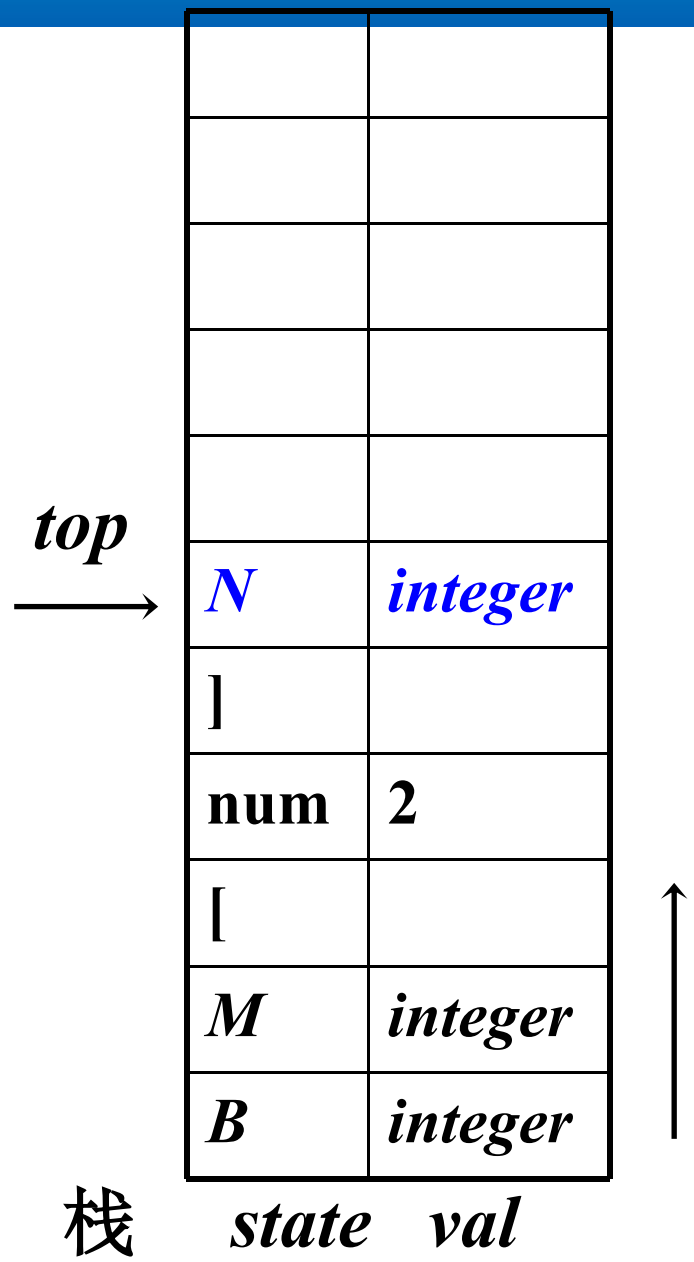
$B \rightarrow$ float $\{B.t = float\}$

$C \rightarrow$ [num] $N\ C_1$

$\{C.t = array(num.val, C_1.t);\ \}$

$N \rightarrow \varepsilon\ \{N.t = C.b\}$

$C \rightarrow \varepsilon\ \{C.t = C.b\}$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

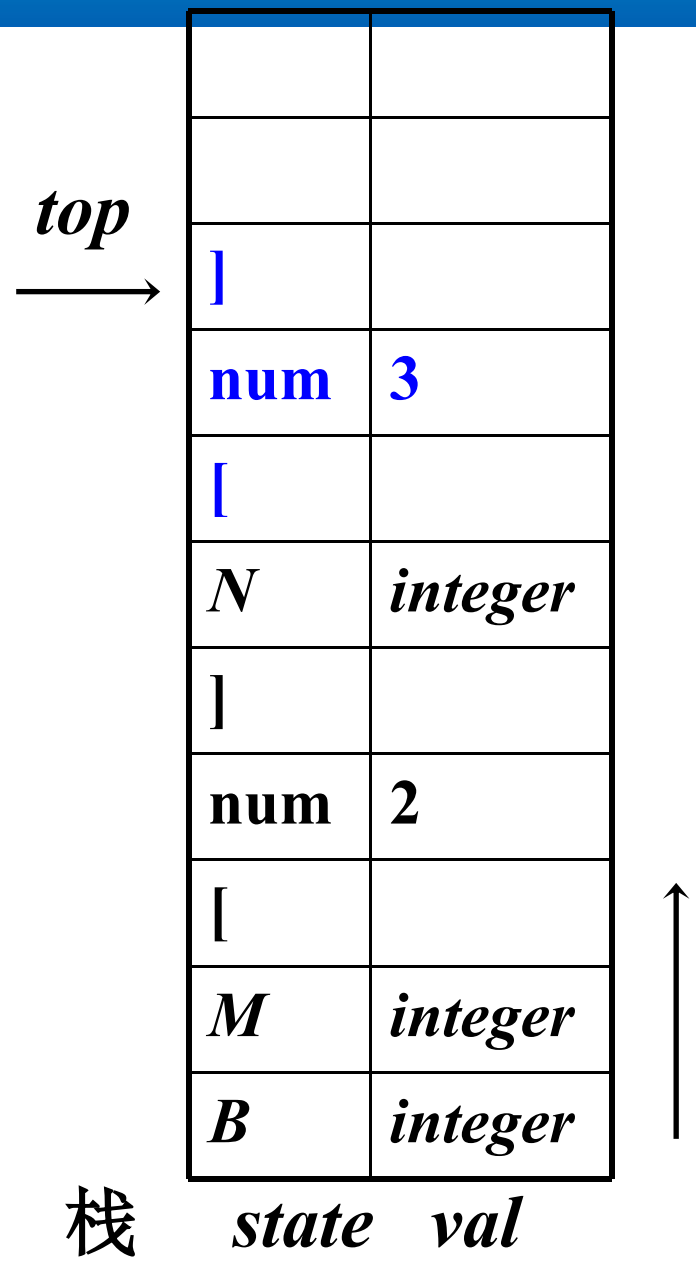
$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

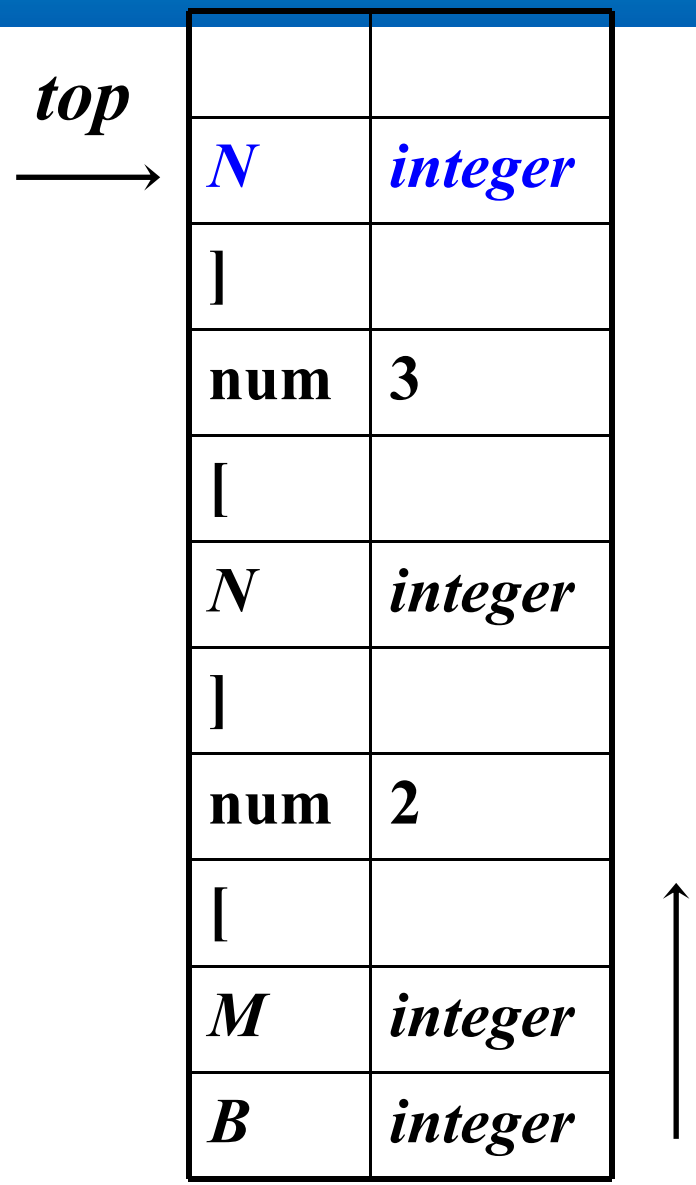
$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

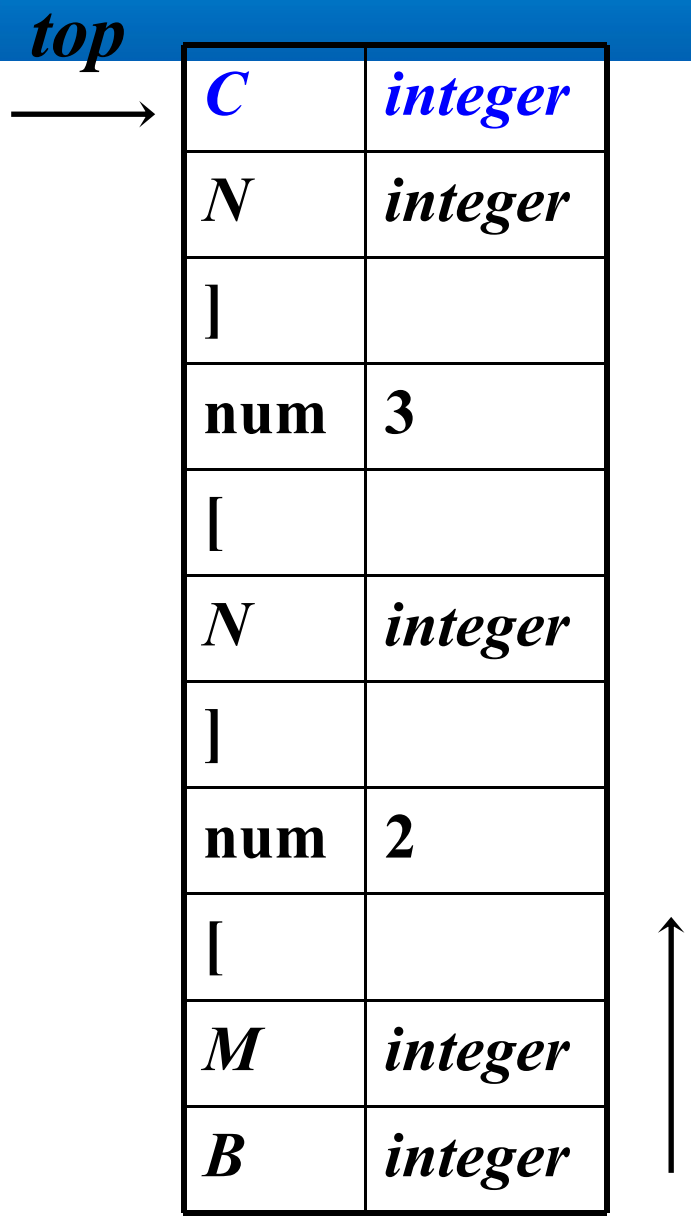
$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}

完成第一次归约

$C \rightarrow$ [num] $N\ C_1$

top →

C	*array(3, integer)*
N	*integer*
]	
num	2
[	
M	*integer*
B	*integer*

栈 *state* *val*

构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

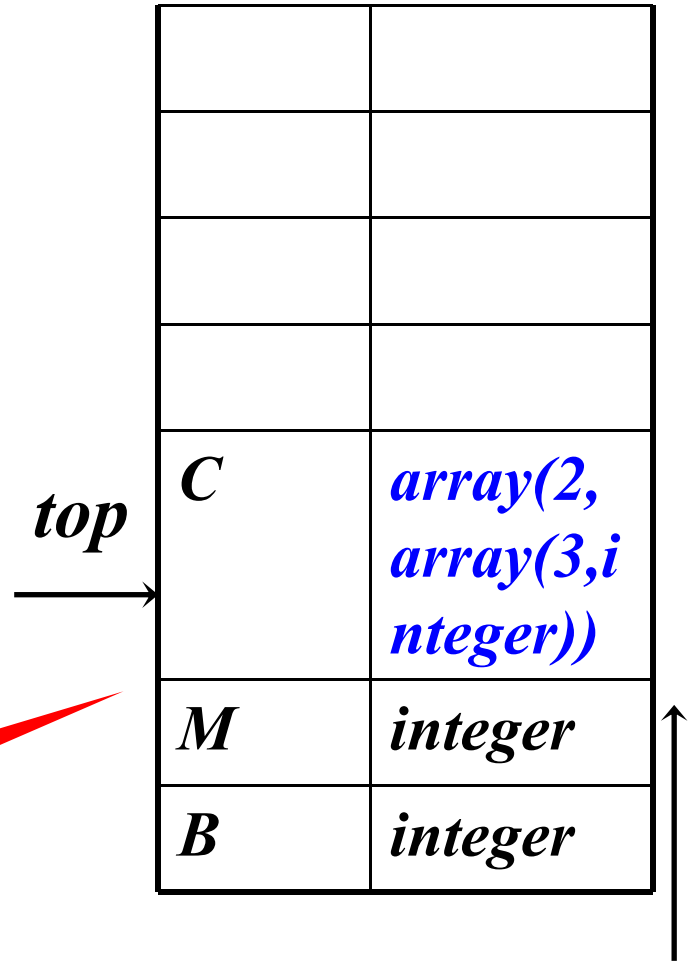
{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}

完成第二次归约

$C \rightarrow$ [num] $N\ C_1$



构造类型表达式的SDT

- 分析int[2][3]的LR栈操作

$T \rightarrow B\ M\ C$ {$T.t = C.t$; }

$M \rightarrow \varepsilon$ {$M.t = B.t$}

$B \rightarrow$ int {$B.t = integer$}

$B \rightarrow$ float {$B.t = float$}

$C \rightarrow$ [num] $N\ C_1$

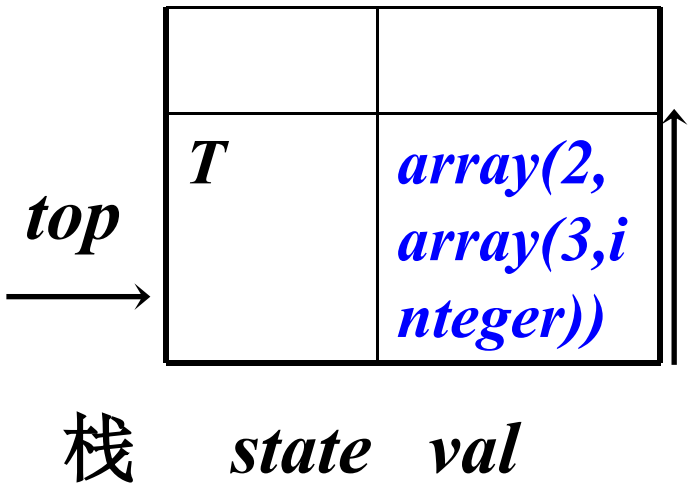
{$C.t = array(num.val, C_1.t)$; }

$N \rightarrow \varepsilon$ {$N.t = C.b$}

$C \rightarrow \varepsilon$ {$C.t = C.b$}

完成第三次归约

$T \rightarrow B\ M\ C$



一起努力
打造国产基础软硬件体系！

李 诚

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年04月03日