数组与广义表

本章主要内容：

- 数组及其运算
- 数组的顺序存储结构
- 矩阵的压缩存储
  - 特殊矩阵：三角矩阵、对称矩阵
  - 稀疏矩阵：三元组顺序表、十字链表
- 广义表

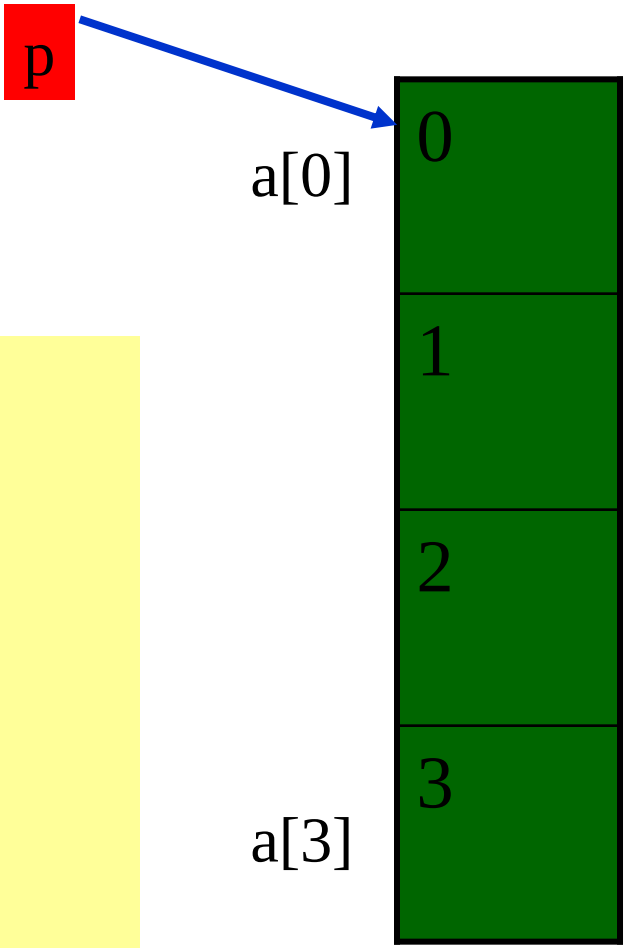
一维数组

int a[4];

a[i]
*(a+i)

a+i的含义：a[i]的地址

p=a;
int b = *(p++);
int c = (*p)++

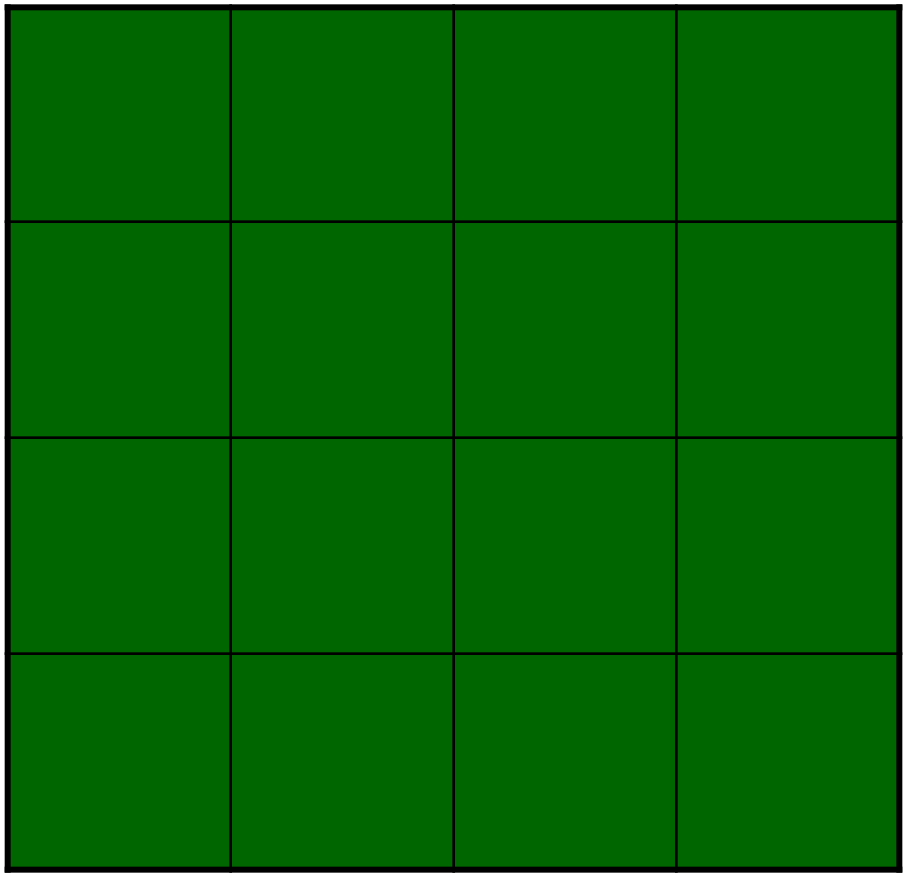


元素个数，存储次序，a[i]偏移：i

二维数组

int a[4][4];

a[0][0]



a[3][0]

a[i][j]
*(a[i]+j)
((a+i)+j)

a+i的含义：a[i]的地址

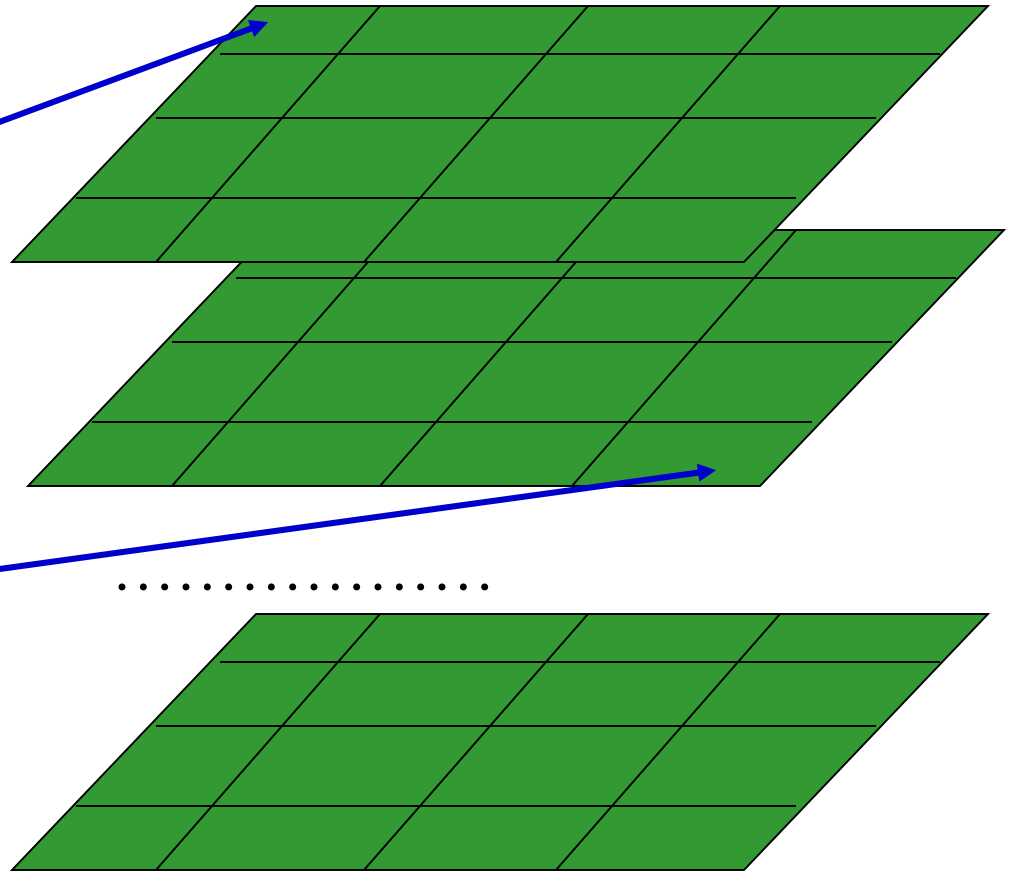
元素个数，存储次序，**a[i][j]偏移：i * 4 + j**

行数　列数

高维数组

int a[5][4][4];

a[0][0][0]



a[1][3][3]

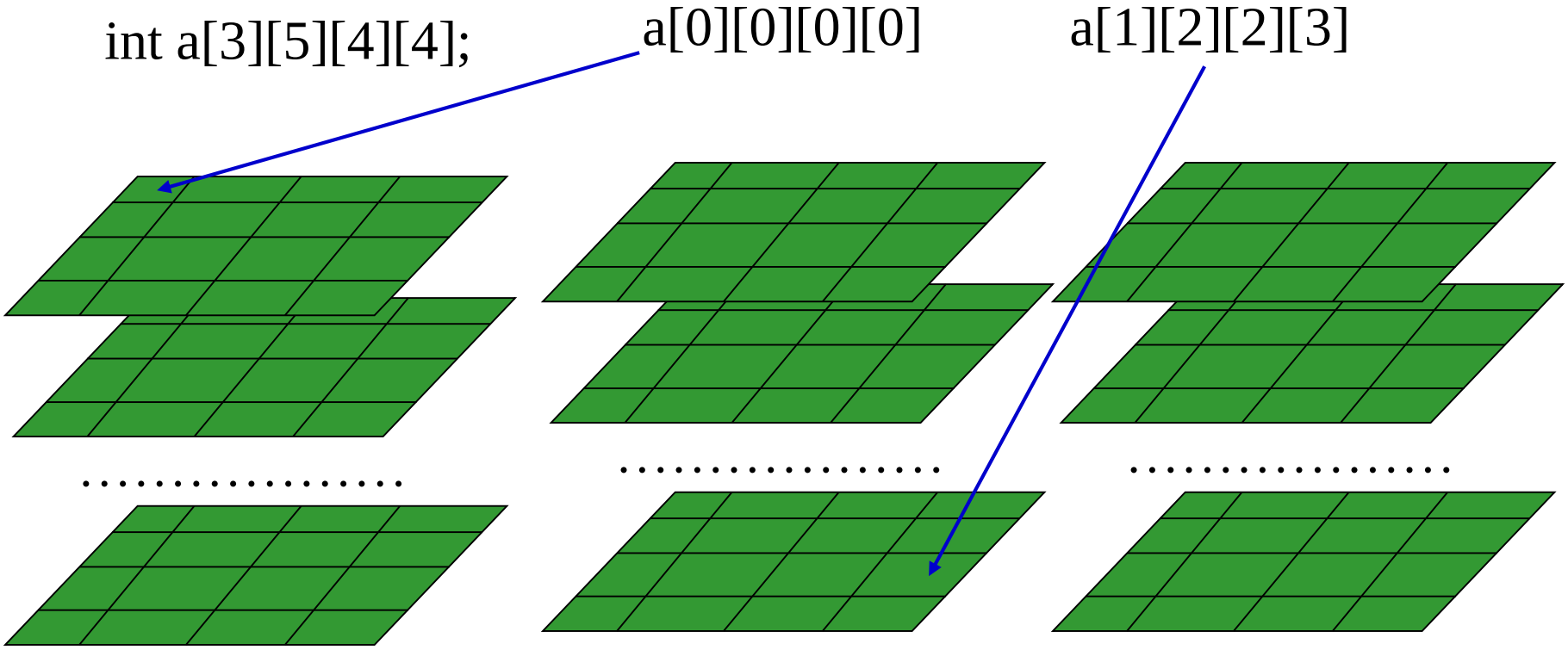
a[4][2][2]

元素个数，存储次序，**a[n][i][j]偏移：n * 4 * 4 + i * 4 + j**

行数　列数

高维数组

int a[3][5][4][4];



a[m][n][i][j]偏移： m * 5 *4 * 4 + n * 4 * 4 + i * 4 + j

层数 行数 列数

1. 数组及其运算

$$A_{mxn}=\begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0n-1} \\ a_{10} & a_{11} & \cdots & a_{1n-1} \\ \cdot & \cdot & & \\ \cdot & \cdot & & \\ a_{m-10} & a_{m-12} & \cdots & a_{m-1n-1} \end{bmatrix}$$

$$A_{mxn}=\begin{bmatrix} \begin{pmatrix} a_{00} \\ a_{10} \\ \cdot \\ \cdot \\ a_{m-10} \end{pmatrix} & \begin{pmatrix} a_{01} \\ a_{11} \\ \cdot \\ \cdot \\ a_{m-12} \end{pmatrix} & \begin{matrix} \cdots \\ \cdots \\ \\ \\ \cdots \end{matrix} & \begin{pmatrix} a_{0n-1} \\ a_{1n-1} \\ \\ \\ a_{m-1n-1} \end{pmatrix} \end{bmatrix}$$

$$A_{mxn}=\begin{bmatrix} (a_{00} \quad a_{01} \quad \cdots \quad a_{0n-1}) \\ (a_{10} \quad a_{11} \quad \cdots \quad a_{1n-1}) \\ \cdot \qquad \cdot \\ \cdot \qquad \cdot \\ (a_{m-10} \quad a_{m-12} \quad \cdots \quad a_{m-1n-1}) \end{bmatrix}$$

这样二维数组转换为线性表结构，每个元素本身又是一个线性表。两种不同的构造得出数组的两种不同的存储结构。

多维问题：三维可以看成多个二维构成的线性表；n维可以看成有多个n-1维构成的线性表。

数组运算：只有存取、修改运算。

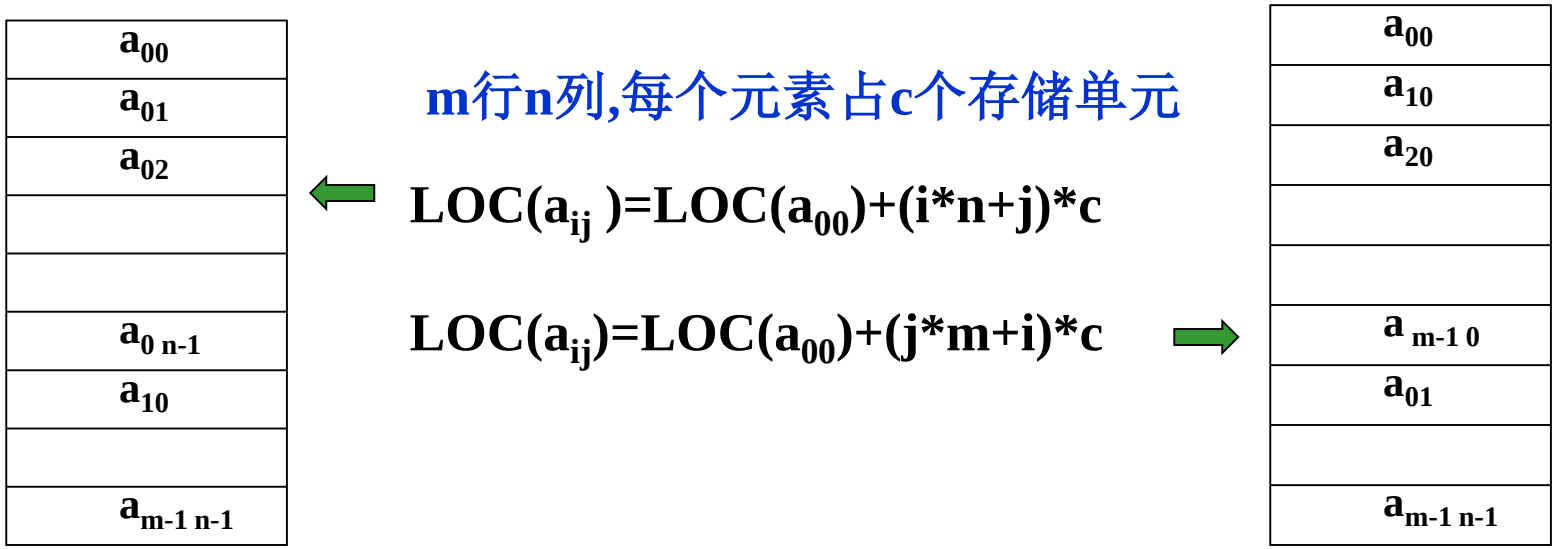
2. 数组的顺序存储结构

由于数组的运算只有存取和修改操作，因此数组一般采用顺序存储结构，即：一片地址连续的存储空间存储数组中的元素。但是，由于数组是多维结构，而存储单元是一维结构，这样必须按照某种次序将数组元素排成一个线性序列。

例如：二维数组既可以看作是由每一行的所有元素作为一个元素构成的一维数组，也可以看作是由每一列的所有元素作为一个元素构成的一维数组。从而导致两种不同的存储方式：行优先和列优先的存储方式。

行优先：C,PASCAL等，先存储完一行中所有元素，再存储下一行。

列优先：FORTRAN等，先存储完一列中所有元素，再存储下一列。



三维数组定位：

行优先： $LOC(a_{kij}) = LOC(a_{000})+(k*m*n+i*n+j)*c$

列优先： $LOC(a_{kij}) = LOC(a_{000})+(k*m*n+j*m+i)*c$

3. 矩阵的压缩存储

二维数组存储矩阵。

特殊矩阵：矩阵中具有相同值的元素的排列具有规律。

稀疏矩阵：矩阵中具有相同值的元素的排列没有规律。

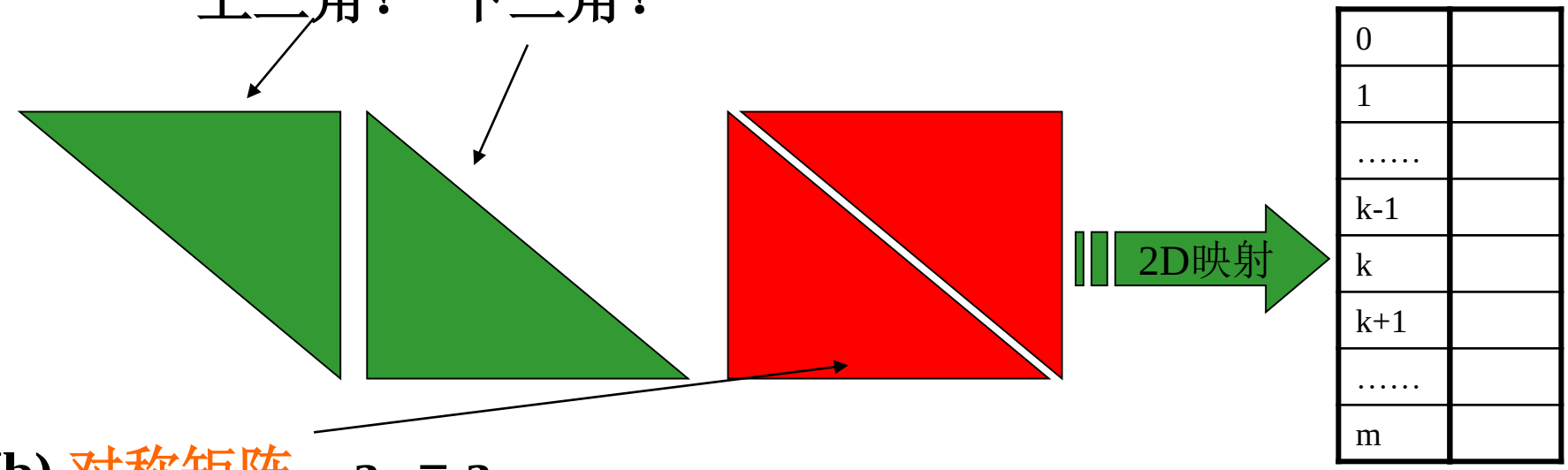
(1)特殊矩阵的压缩存储

(a) 三角矩阵：用n(n+1)/2+1个单元存储构成的线性表存储。

给定行列，如何确定在线性表中的位置？

给定线性表中的一个位置，如何确定行列？

上三角？　下三角？



(b) 对称矩阵：$a_{ij} = a_{ji}$

存储：有n(n+1)/2个单元构成的线性表；

定位：给定行列，如何确定在线性表中的位置？

给定线性表中的一个位置，如何确定行列？

(2)稀疏矩阵的压缩存储

定义：矩阵中非空元素个数远远少于矩阵元素个数，
并且空元素排列无规律。

存储：存储非空元素的同时，存储位置辅助信息：(i, j, a_{ij})
(i, j, a_{ij})称为一个三元组，唯一表达一个非空元素。

存储方式：这些三元组如何组织，方便矩阵的运算？

a) 三元组顺序表：三元组构成的顺序表作为稀疏矩阵的压缩存储方式，便于矩阵的转置、相乘等操作。

```
#define  maxsize
 struct node
 {    int i, j;
      datatype  v;
 };
```

```
struct matrix
{
     int mu, nu, tu;
     struct node data[maxsize];
};
```

15	0	0	22	0	-15
0	11	3	0	0	0
0	0	0	-6	0	0
0	0	0	0	0	0
91	0	0	0	0	0
0	0	28	0	0	0

	行	列	值
0	0	0	15
1	0	3	22
2	0	5	-15
3	1	1	11
4	1	2	3
5	2	3	-6
6	4	0	91
7	5	2	28

给定行列(i、j)检索数值：
ij有序，因此可以使用快速的二分法检索：$O(\log_2(k))$
k是非零元素个数。

给定数值，检索行列：顺序检索。

b) 伪地址表示法

【伪地址(偏移量)，数值】构成二元组

由行列计算伪地址：IA = i*n+j，有序。

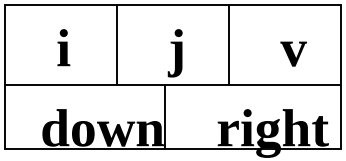
由伪地址计算行列：i = IA/n, j = IA%n。

伪地址有序，可以折半检索：$O(\log_2(k))$

	伪地址	值
0	0	15
1	3	22
2	5	-15
3	7	11
4	8	3
5	15	-6
6	24	91
7	32	28

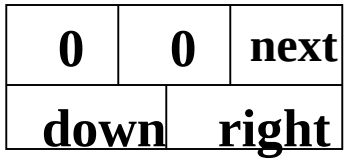
c) 十字链表：三元组构成的链表作为稀疏矩阵的压缩存储方式，便于矩阵的加减等非空元素数目变化大的运算。

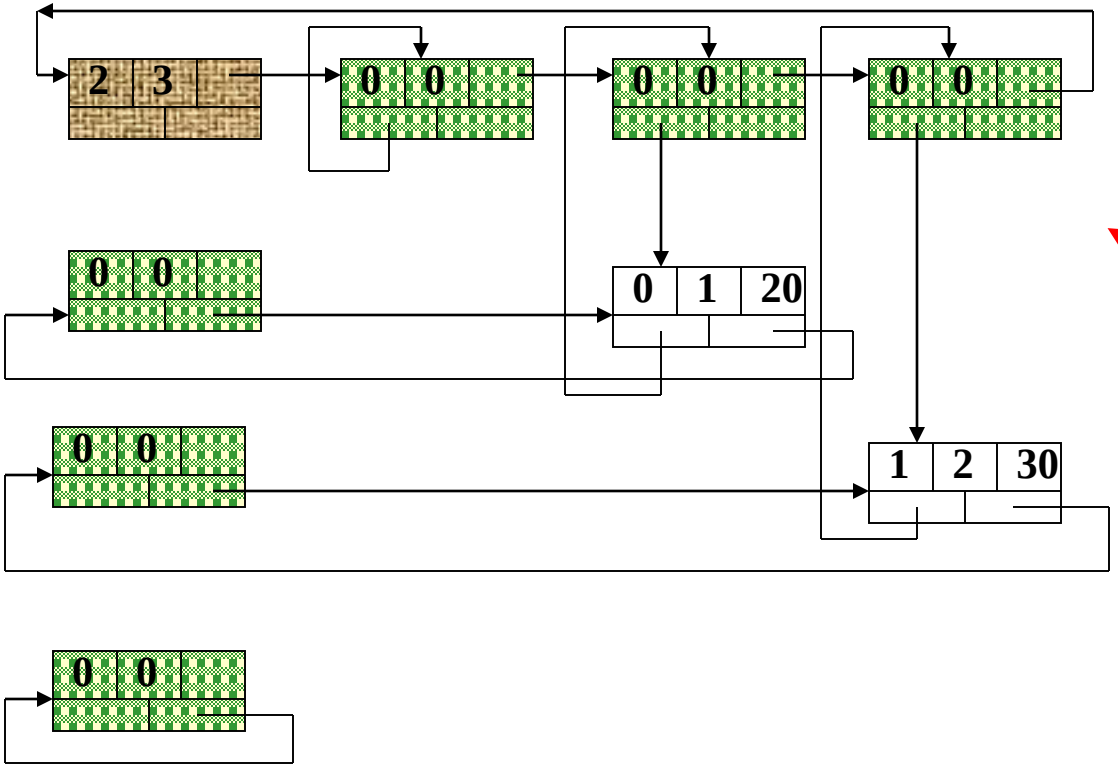
结点结构



(i,j,v)三元组表达一个非空元素。
Down指针连接同列的下一个非空元素；
Right指针连接同行的下一个非空元素。

每一行增加一个行表头结点，每一列增加一个列表头结点：





0	20	0
0	0	30
0	0	0

4. 广义表

$L=(a_1, a_2, \ldots, a_i, \ldots, a_n)$，其中$a_i$可以是不可分割的原子元素，也可以是一个广义表。

A=()
L=(a,b)
C=(a,(b,c,d))
E=(a,E) = (a, (a, (a,(…))))

atom	data/snext	next

结点结构

存储结构：只能采用动态链表存储。

两种不同结点：原子结点和子表结点

atom = 1：结点为原子元素，此时**data**存放元素值；

0：结点为子表，此时**snext**指向子表。

书写中约定：用大写字母表示广义表，用小写字母表示原子。例如：

E＝()

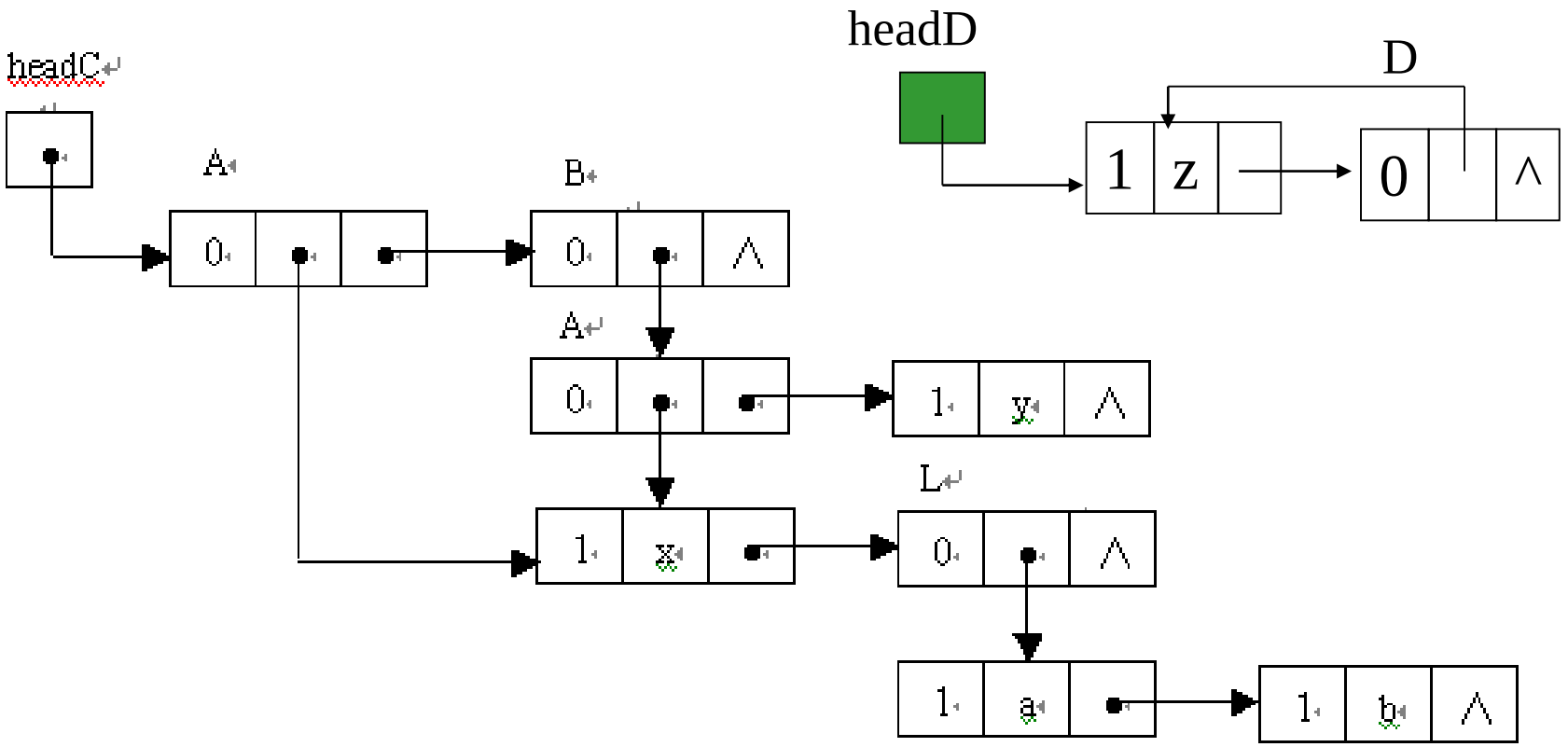
L＝(a , b)

A＝(x , L)＝(x ,(a , b))

B＝(A , y)＝((x ,(a , b)), y)

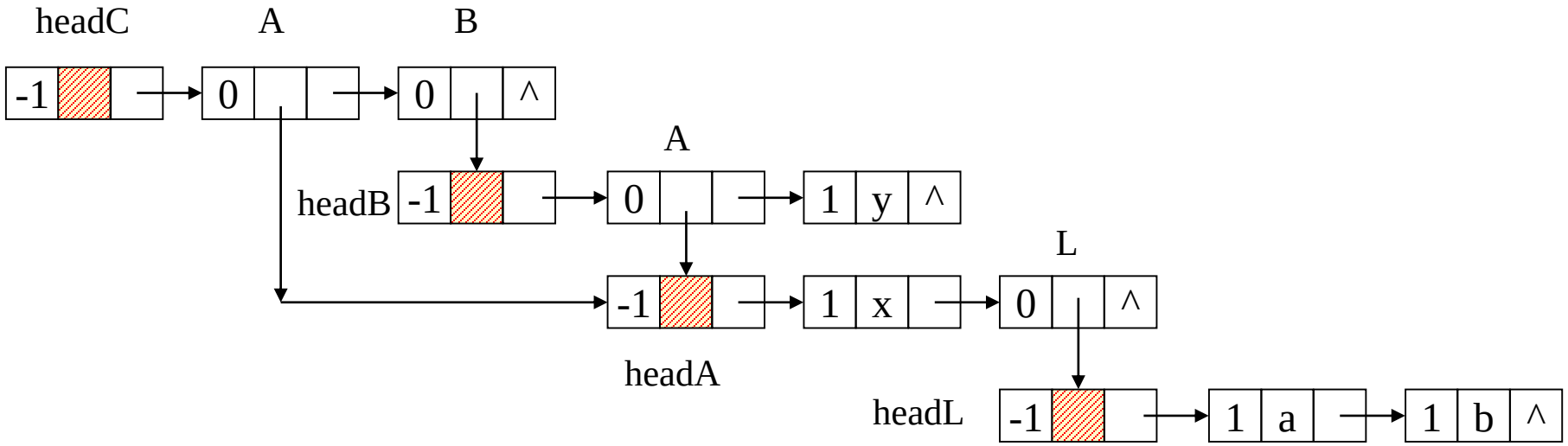
C＝(A , B)＝((x ,(a , b)),((x ,(a , b)), y))

D＝(z , D)＝(z ,(z ,(z ...)))

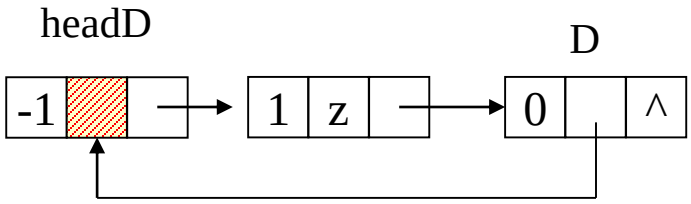


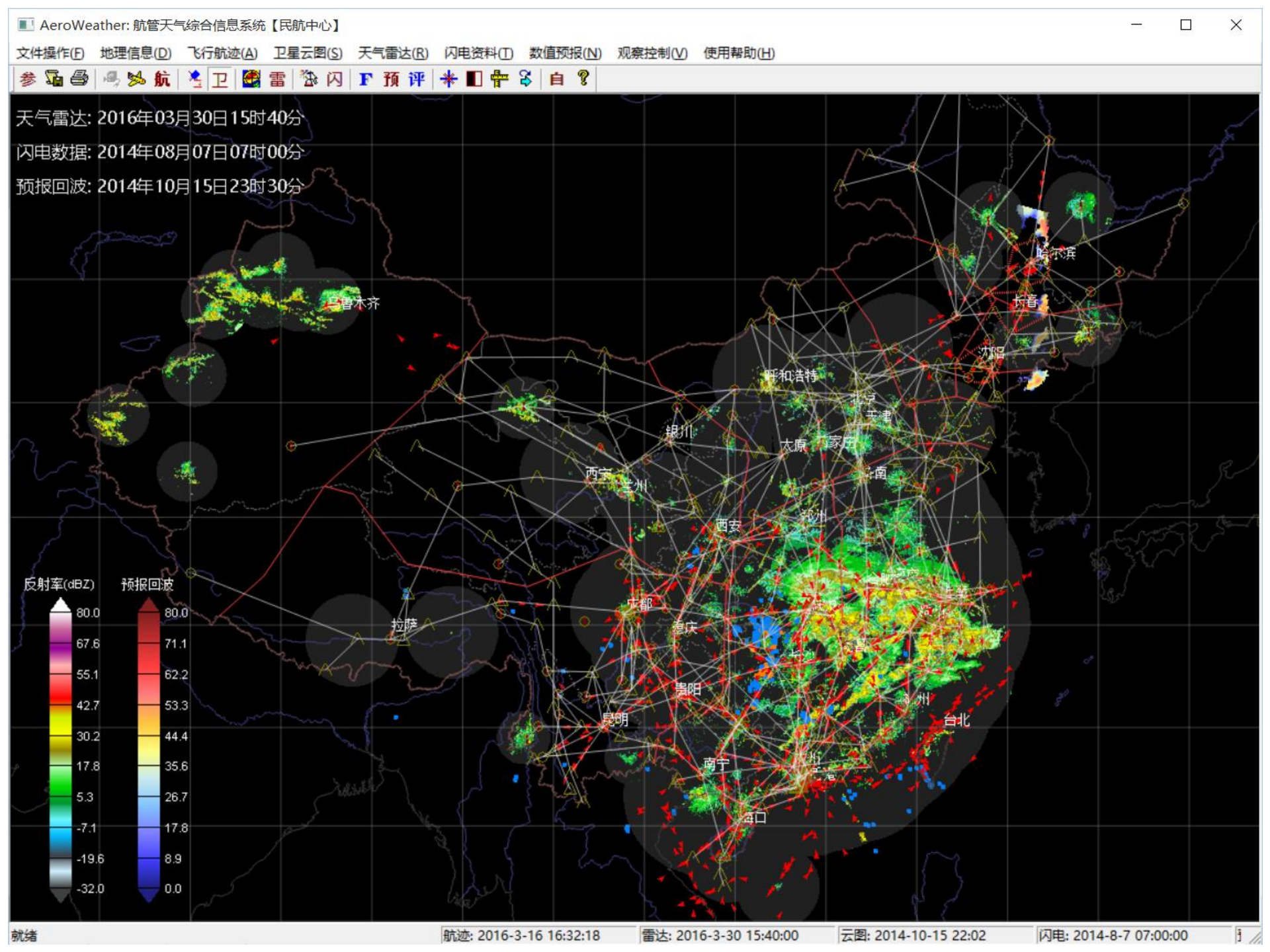
一般的单链表存储：既有原子结点，又有子表结点

子表带头结点的单链表存储：原子结点不共享



删除原子结点：在单链表中，删除运算即可




AeroWeather: 航管天气综合信息系统【民航中心】
文件操作(F) 地理信息(D) 飞行航迹(A) 卫星云图(S) 天气雷达(R) 闪电资料(T) 数值预报(N) 观察控制(V) 使用帮助(H)
天气雷达: 2016年03月30日15时40分
闪电数据: 2014年08月07日07时00分
预报回波: 2014年10月15日23时30分
乌鲁木齐
哈尔滨
长春
沈阳
呼和浩特
北京
天津
银川
太原
石家庄
西宁
兰州
南京
郑州
西安
成都
拉萨
重庆
贵阳
昆明
台北
南宁
海口
反射率(dBZ)
80.0
67.6
55.1
42.7
30.2
17.8
5.3
-7.1
-19.6
-32.0
预报回波
80.0
71.1
62.2
53.3
44.4
35.6
26.7
17.8
8.9
0.0
就绪
航迹: 2016-3-16 16:32:18
雷达: 2016-3-30 15:40:00
云图: 2014-10-15 22:02
闪电: 2014-8-7 07:00:00

全国天气雷达拼图：

1）采用等经纬度间隔矩形拼接，实质上就是二维网格；

2）雷达布局不均匀，造成很多无效区域（点）；

3）存在降水回波的区域是“稀疏区域”；

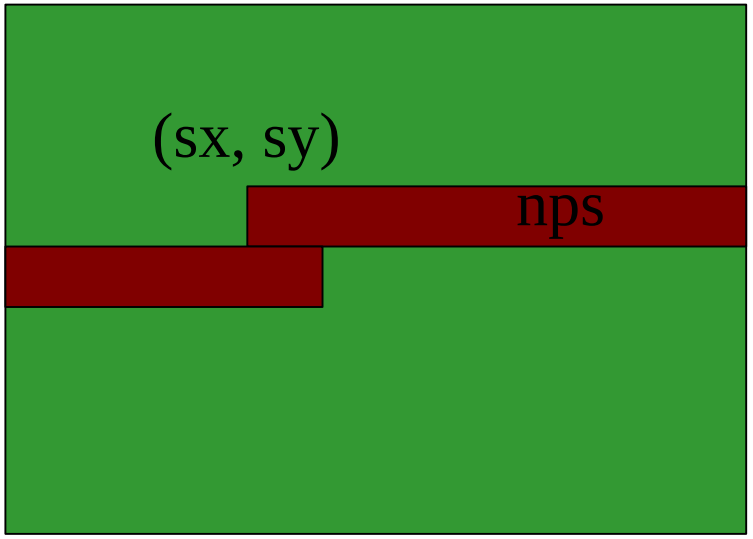
因此采用稀疏矩阵压缩存储（类三元组）。

```
TPOINT
{
    int sx, sy;          //起始坐标（行列）
    int nps;             //连续nps个有效点
    float *ps;           //nps个有效点的数值（数组）

}
{
    float lon1, lon2, lat1, lat2;
    int rows, cols;
    TPOINT *points;
}
```

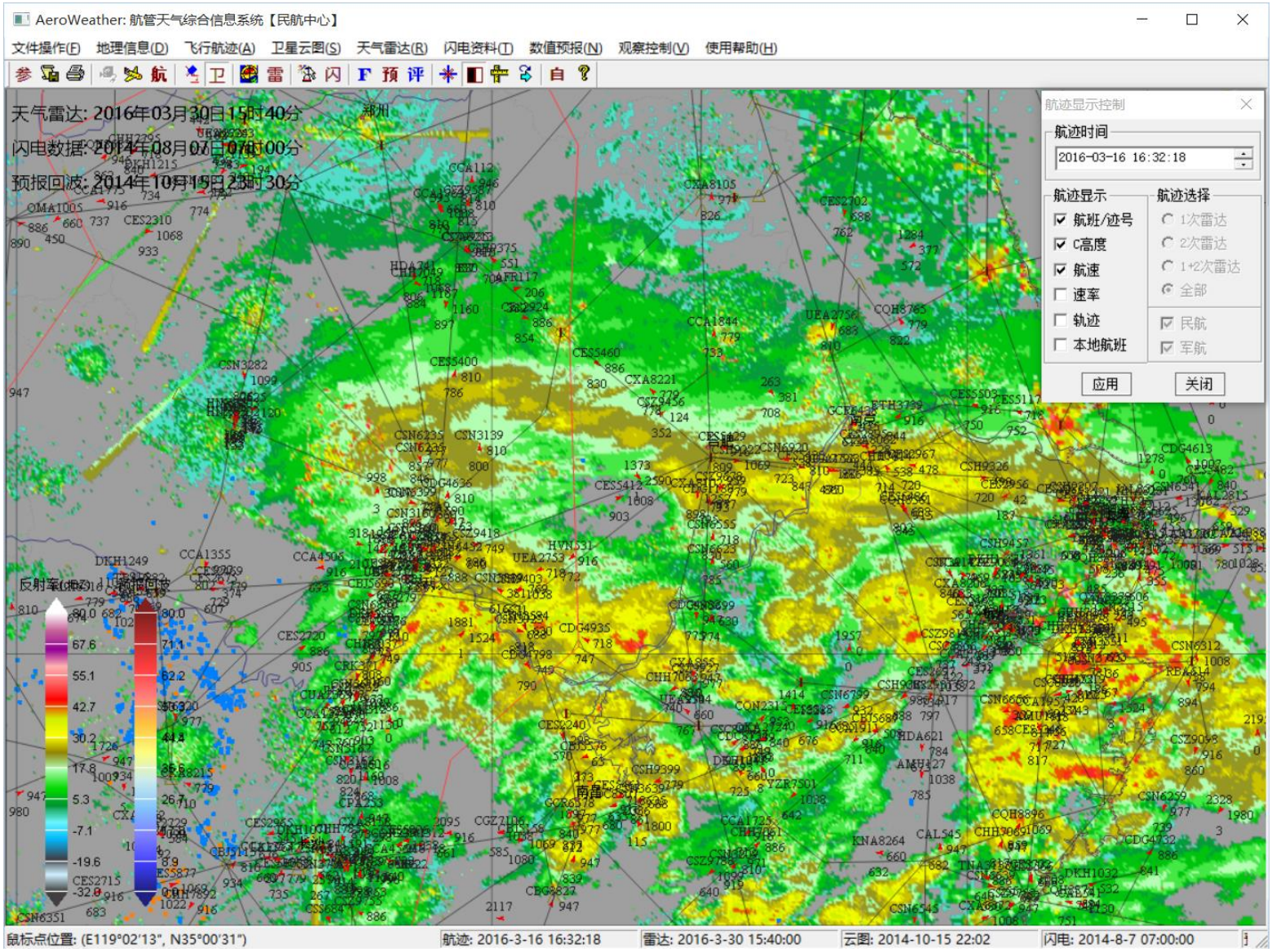
问题1：如何恢复二维数组（矩阵）？

```
Pickup_TPOINT(TPOINT *pTP)
{
    //grid2d[rows][cols];
    int k, m;
    int k = pTP->sy*cols+sx;
    int m;
    for (m = 0; m < pTP->nps; m++)
    {
        grid2d[k+m] = pTP->ps[m];
    }
}
```



问题2：如何由二维网格压缩存储？

（找到一个有效点，连续nps个有效网格点形成一个TPOINT）



不考虑飞机自身问题情况下，与航空危险天气相关的其它问题：（航路选择，航路流量控制，绕飞策略……）