Princípios de Desenvolvimento de Algoritmos MAC122

Prof. Dr. Paulo Miranda
IME-USP

Matrizes esparsas

Matrizes esparsas

- Uma matriz é considerada esparsa quando a grande maioria dos seus elementos são nulos, não havendo uma regra simples para identificação dos elementos não nulos.
- Solução clássica:
  - Implementação através de um conjunto de listas ligadas que apontam para elementos não nulos.
  - Elementos que possuem valor zero não são armazenados.

Matrizes esparsas

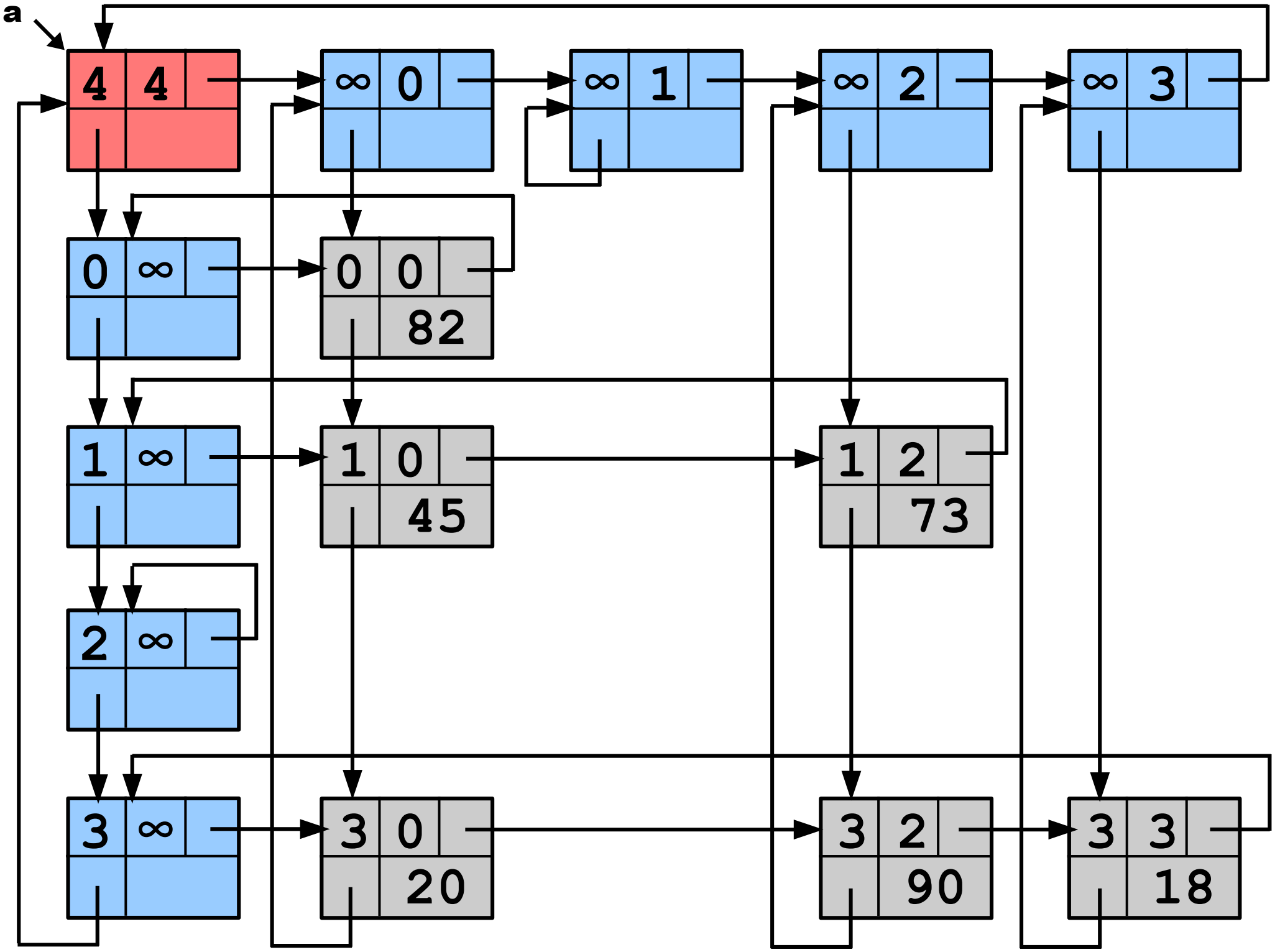
- Definição típica da estrutura utilizada.

```
typedef struct _RegEsparsa{
  float valor;
  int   linha;
  int   coluna;
  struct _RegEsparsa *direita;
  struct _RegEsparsa *abaixo;
} RegEsparsa;


typedef RegEsparsa* MatrizEsparsa;


/*'MatrizEsparsa' é um tipo abstrato que
   está sendo implementado. */
```

Matrizes esparsas

- Exemplo:

$$\begin{pmatrix} 82 & 0 & 0 & 0 \\ 45 & 0 & 73 & 0 \\ 0 & 0 & 0 & 0 \\ 20 & 0 & 90 & 18 \end{pmatrix}$$


a
4 4
∞ 0
∞ 1
∞ 2
∞ 3
0 ∞
0 0
82
1 ∞
1 0
45
1 2
73
2 ∞
3 ∞
3 0
20
3 2
90
3 3
18

Matrizes esparsas

- Criação da estrutura.

```
RegEsparsa* AlocaRegEsparsa(){
  RegEsparsa* q;
  q = (RegEsparsa*)calloc(1, sizeof(RegEsparsa));
  if(q==NULL) exit(-1);
  return q;
}
```

Matrizes esparsas

```
#include <limits.h>
/*Cria uma matriz nula, com m linhas
  e n colunas.*/
MatrizEsparsa CriaMatrizEsparsa(int m,
                                int n){
  MatrizEsparsa a;
  RegEsparsa *p,*pp;
  int k;
  a = AlocaRegEsparsa();
  a->linha  = m;
  a->coluna = n;
  a->abaixo  = a;
  a->direita = a;

  pp = a;
  for(k = 0; k < n; k++){
    p = AlocaRegEsparsa();
    p->linha  = INT_MAX;
    p->coluna = k;
    p->abaixo  = p;
    p->direita = pp->direita;
    pp->direita = p;
    pp = p;
  }

  pp = a;
  for(k = 0; k < m; k++){
    p = AlocaRegEsparsa();
    p->linha  = k;
    p->coluna = INT_MAX;
    p->direita = p;
    p->abaixo  = pp->abaixo;
    pp->abaixo = p;
    pp = p;
  }

  return a;
}
```

Matrizes esparsas

```
void AtribuiMEsparsa(MatrizEsparsa a,
                     int i, int j,
                     float x){
  RegEsparsa *p,*q,*pp,*qq,*r;
  int k;

  p = a;
  q = a;

  for(k = 0; k <= i; k++)
    p = p->abaixo;

  for(k = 0; k <= j; k++)
    q = q->direita;

  do{
    pp = p;
    p  = p->direita;
  }while(p->coluna < j);

  do{
    qq = q;
    q  = q->abaixo;
  }while(q->linha < i);

  if(p->coluna == j){
    if(x!=0.0) p->valor = x;
    else{ /*Remove elemento:*/
      qq->abaixo  = p->abaixo;
      pp->direita = p->direita;
      free(p);
    }
  }
  /*Insere elemento:*/
  else if(x!=0.0){
    r = AlocaRegEsparsa();
    r->coluna = j;
    r->linha  = i;
    r->valor  = x;
    r->abaixo  = qq->abaixo;
    r->direita = pp->direita;
    qq->abaixo  = r;
    pp->direita = r;
  }
}
```

Matrizes esparsas

- Consultar posição **(i,j)** da matriz.

```
float ValorMatrizEsparsa(MatrizEsparsa a,
                         int i, int j){
  RegEsparsa* p;
  int k;

  p = a;
  for(k = 0; k <= i; k++) p = p->abaixo;

  do{
    p = p->direita;
  }while(p->coluna < j);

  if(p->coluna==j) return p->valor;
  else             return 0.0;
}
```

Matrizes esparsas

- Imprimindo a matriz esparsa.

```
void  ImprimeMatrizEsparsa(MatrizEsparsa a){
  RegEsparsa *p,*q;
  int k,j;

  p = a;
  for(k=1; k<=a->linha; k++){
    p = p->abaixo;
    q = p->direita;
    j = 0;
    while(q!=p){
      for(; j<q->coluna; j++) printf(" %6.2f ",0.0);
      printf(" %6.2f ",q->valor); j++;
      q = q->direita;
    }
    for(; j < a->coluna; j++) printf(" %6.2f ",0.0);
    printf("\n");
  }
}
```

Matrizes esparsas

- Exemplo de função principal.

```
int main(){
  MatrizEsparsa a;
  int m,n;

  m = 4;
  n = 4;
  a = CriaMatrizEsparsa(m, n);

  AtribuiMEsparsa(a, 1, 2, 73);
  AtribuiMEsparsa(a, 3, 2, 90);
  AtribuiMEsparsa(a, 3, 0, 20);
  AtribuiMEsparsa(a, 0, 0, 82);
  AtribuiMEsparsa(a, 1, 0, 45);
  AtribuiMEsparsa(a, 3, 3, 18);

  ImprimeMatrizEsparsa(a);

  return 0;
}
```