

Aplicações de listas e outras estruturas

Baseado no material de Thiago A. S. Pardo

Algoritmos e Estruturas de Dados I

Debora Medeiros

Grandes números

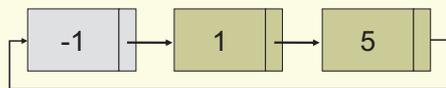
- **Problema:** lidar com números muito grandes
- Em C, inteiros (mesmo long int) são limitados
- Como somar números inteiros maiores do que o tamanho do tipo permite?
 - Listas!

2

Grandes números

- Representando números como listas

- 15



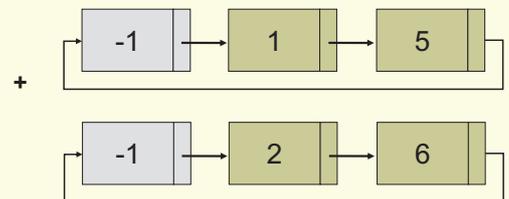
3

Grandes números

- Soma de dois números

- Bloco somados dois a dois, da direita para a esquerda

- Exemplo: 15+26

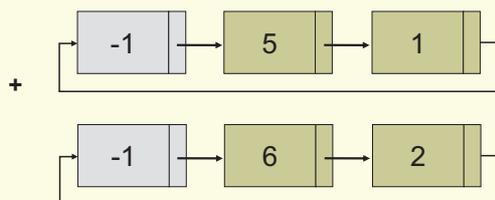


4

Grandes números

- Para facilitar nossa vida, números já são representados ao contrário

- Exemplo: 15+26



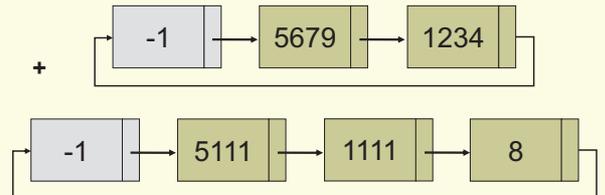
5

Grandes números

- Como os números tratados por esse mecanismo são muito grandes, pode-se aproveitar melhor o tipo inteiro: uso otimizado de memória

- Exemplo: 12.345.679 + 811.115.111

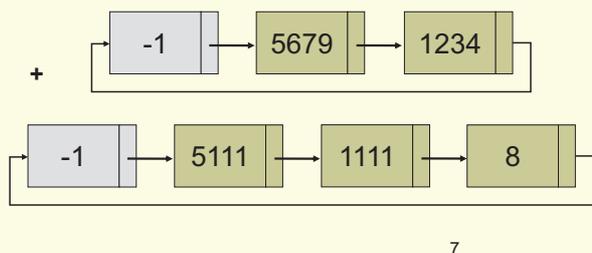
- Produz-se uma outra lista como resultado



6

Grandes números

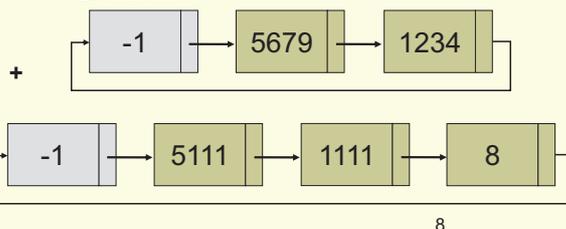
- Como recuperar o número somado para colocar na nova lista?
- Como recuperar o “sobe 1”?



7

Grandes números

- Como recuperar o número somado para colocar na nova lista?
 - Soma % 10.000 (por que 4 zeros?)
- Como recuperar o “sobe 1”?
 - Soma / 10.000



8

Grandes números

Exercício para casa

- Implemente em C uma sub-rotina para somar dois grandes números utilizando uma lista circular com nó de cabeçalho
 - As duas listas a serem somadas devem ser passadas por parâmetros, sendo que o ponteiro para a nova lista contendo a soma deve ser retornado em um outro ponteiro (por parâmetro também).

9

Matrizes

- Matriz é um arranjo (tabela) retangular de números dispostos em linhas e colunas

$$A \begin{bmatrix} 1 & 0 & 4 & -3 \\ 2 & 5 & 3 & 4 \\ 9 & 8 & -2 & 1 \end{bmatrix} \quad B \begin{bmatrix} 3 & 7 & 4 \\ 1 & 0 & 6 \\ 9 & 2 & 8 \end{bmatrix}$$

nº de elementos = nº de linhas * nº de colunas

Matriz = Arranjo bidimensional

10

Matrizes especiais

$$A \begin{bmatrix} 1 & 0 & 0 \\ 2 & 3 & 0 \\ 4 & 5 & 6 \end{bmatrix}$$

Triangular inferior

$$B \begin{bmatrix} 1 & 2 & 0 & 0 \\ 2 & 4 & 5 & 0 \\ 0 & 6 & 7 & 8 \\ 0 & 0 & 9 & 10 \end{bmatrix}$$

Tri-diagonal

$$C \begin{bmatrix} 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Matriz esparsa: excessivo nº de elementos nulos (0)

11

Matrizes esparsas

$$C \begin{bmatrix} 1 & 0 & 0 & 3 & 0 & 0 & 0 & \dots & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & -1 & 4 & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & \dots & 0 \end{bmatrix}$$

700 x 900 = 630.000 elementos

Matriz esparsa com 9 elementos **não nulos**

12

Matrizes esparsas

■ Uso da **matriz tradicional**

■ **Vantagem**

- Ao se representar dessa forma, preserva-se o **acesso direto** a cada elemento da matriz
 - Algoritmos simples

■ **Desvantagem**

- Muito espaço para armazenar zeros

13

Matrizes esparsas

■ Necessidade

- **Método alternativo** para representação de matrizes esparsas

■ **Solução**

- Estrutura de **lista encadeada** contendo somente os **elementos não nulos**

14

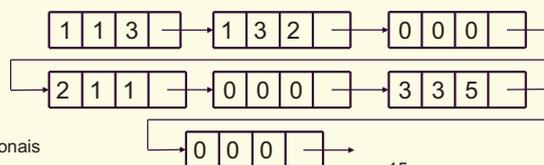
Matrizes esparsas - solução 1

■ Listas simples encadeadas

linha	coluna	valor	prox
-------	--------	-------	------

Estrutura de um nó:

- linha, coluna: posição
- valor: ≠ zero
- prox: próximo nó



15

Matrizes esparsas - solução 1

■ **Desvantagens**

- Perda da natureza bidimensional de matriz
- Acesso ineficiente à linha
 - Para acessar o elemento na i-ésima linha, deve-se atravessar as i-1 linhas anteriores
- Acesso ineficiente à coluna
 - Para acessar os elementos na j-ésima coluna, tem que se passar por várias outras antes

■ **Questão**

- Como organizar essa lista, preservando a natureza bidimensional de matriz?

16

Matrizes esparsas - solução 2

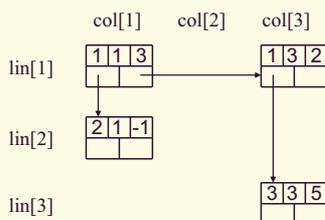
■ **Listas cruzadas**

- Para cada matriz, usam-se dois vetores com N ponteiros para as linhas e M ponteiros para as colunas

$$A_{3 \times 3} = \begin{bmatrix} 3 & 0 & 2 \\ -1 & 0 & 0 \\ 0 & 0 & 5 \end{bmatrix}$$

Estrutura de um nó:

linha	coluna	valor
proxlin	proxcol	



17

Matrizes esparsas - solução 2

■ Listas cruzadas

- Cada elemento não nulo é mantido **simultaneamente em duas listas**
 - Uma para sua linha
 - Uma para sua coluna

18

Matrizes esparsas

- Listas cruzadas vs. matriz tradicional
 - Em termos de espaço
 - Supor que inteiro e ponteiro para inteiro ocupam um bloco de memória
 - Listas cruzadas: tamanho do vetor de linhas (nl) + tamanho do vetor de colunas (nc) + n elementos não nulos * tamanho do nó
 - $nl+nc+5n$
 - Matriz tradicional bidimensional
 - $nl*nc$

19

Matrizes esparsas

- Listas cruzadas vs. matriz tradicional
 - Em termos de tempo
 - Operações mais lentas em listas cruzadas: acesso não é direto

20

Matrizes esparsas

- Listas cruzadas vs. matriz tradicional
 - Necessidade de avaliação tempo-espaço para cada aplicação
 - Em geral, usam-se listas cruzadas quando no máximo 1/5 dos elementos forem não nulos
 - De onde vem isso?

21

Matrizes esparsas

- Listas cruzadas vs. matriz tradicional
 - Necessidade de avaliação tempo-espaço para cada aplicação
 - Em geral, usam-se listas cruzadas quando no máximo 1/5 dos elementos forem não nulos
 - De onde vem isso?

Dica: $nl+nc+5n < nl*nc$

22

Matrizes esparsas - operações

- Em geral
 - Multiplicar uma dada linha ou coluna por uma constante
 - Somar uma constante a todos os elementos de uma linha ou coluna
 - Somar duas matrizes esparsas de igual dimensão
 - Multiplicar matrizes esparsas
 - Transpor matrizes esparsas
 - Inserir, remover ou alterar elementos
 - Etc.

23

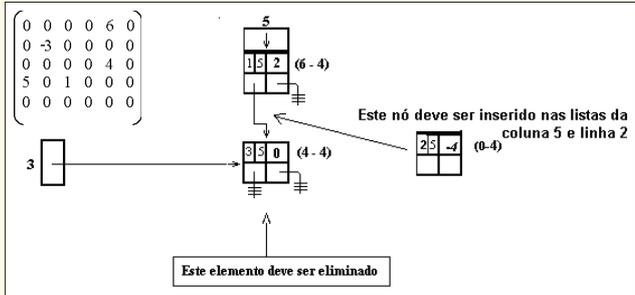
Matrizes esparsas - operações

- Após a realização de alguma operação sobre a matriz
 - Quando um elemento da matriz se torna nulo
 - Remoção do elemento
 - Quando algum elemento se torna não nulo
 - Inserção do elemento

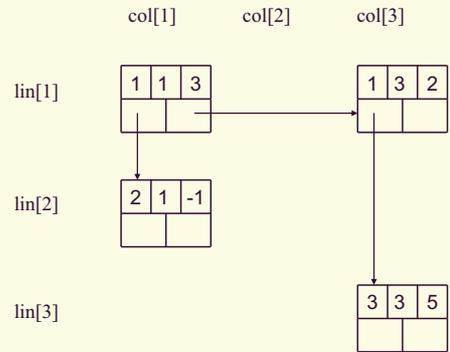
24

Matrizes esparsas - operações

- Por exemplo, ao se somar -4 a coluna 5 do exemplo



Exercício: somar -5 a coluna 3



26

Exercício

- Declare em C a estrutura da matriz esparsa representada via lista cruzada

27

Exercício

- Declare em C a estrutura da matriz esparsa representada via lista cruzada

```
#define n ... //número de linhas
#define m ... //número de colunas
```

```
typedef struct no {
    int linha, coluna, valor;
    struct no *proxlin, *proxcol;
} no;
```

```
typedef struct {
    no *L[n], *C[m];
} MatrizEsparsa;
```

28