Ordenação

Linguagem Haskell

Maria Adriana Vidigal de Lima

Faculdade de Computação - UFU

Dezembro - 2009

1. Ordenação de Valores em Haskell
   - Técnicas de Ordenação
     - Ordenação por Seleção
     - Ordenação por Inserção
     - Ordenação Rápida
     - Ordenação por Bolha

Fundamentos

- Ordenação por Seleção (*Selection Sort*)
- Ordenação por Inserção (*Insertion Sort*)
- Ordenação Rápida (*Quick Sort*)
- Ordenação por Bolha (*Bubble Sort*)

Ordenação por Seleção

Estratégia de ordenação

- Encontrar o menor elemento da lista.
- Inserir o elemento encontrado numa nova lista, e removê-lo da lista de origem.
- Repetir as ações acima até que a lista original seja vazia.

Ordenação por Seleção

Exemplo:

```
Lista Inicial:
  [3,4,5,1,2]
  Menor Elemento: 1
  [1]  --  [3,4,5,2]
           Menor Elemento: 2
           [1,2] -- [3,4,5]
                    Menor Elemento: 3
                    [1,2,3] -- [4,5]
                               Menor Elemento:  4
                               [1,2,3,4] -- [5]
                                            Menor Elemento: 5
                                            [1,2,3,4,5]  -- []
```

Ordenação por Seleção em Haskell

```
selecao:: (Ord a) => [a]->[a]
selecao [] = []
selecao xs = [x] ++ selecao (remove x xs)
                    where x = minimo xs

remove:: (Ord a) => a->[a]->[a]
remove a [] = []
remove a (x:xs)
  | a==x = xs
  | otherwise = x:(remove a xs)

minimo::(Ord a) => [a]->a
minimo [] = undefined
minimo [x] = x
minimo (x:xs)
  | x <= (minimo xs) = x
  | otherwise = minimo xs
```

Ordenação por Inserção

Estratégia de ordenação:

- Percorrer a lista da esquerda para a direita, e para cada novo elemento encontrado, inseri-lo de forma ordenada numa nova lista.
- O processo termina quando a lista de origem for vazia.
- Este tipo de ordenação é rápido para pequenas listas.

Ordenação por Inserção

Exemplo:

```
Lista Inicial: [3,4,2,5,1]

Inserção Ordenada
Lista Original  --- Lista Ordenada
[3,4,2,5,1]     --- []
[4,2,5,1]       --- [3]
[2,5,1]         --- [3,4]
[5,1]           --- [2,3,4]
[1]             --- [2,3,4,5]
[]              --- [1,2,3,4,5]
```

Ordenação por Inserção em Haskell

Implementação usando recursividade simples:

```
insercao ::(Ord a) => [a] -> [a]
insercao  []     = []
insercao  (x:xs) = insereOrd x (insercao xs)

insereOrd ::(Ord a) => a -> [a] -> [a]
insereOrd x [] = [x]
insereOrd x (y:ys)
  | x <= y = (x:y:ys)
  | otherwise = y: (insereOrd x ys)
```

Ordenação por Inserção em Haskell

Implementação usando a função genérica (*foldr*):

```
insercao2 :: Ord a => [a] -> [a]
insercao2 = foldr insereOrd []

insereOrd ::(Ord a) => a -> [a] -> [a]
insereOrd x [] = [x]
insereOrd x (y:ys)
  | x <= y = (x:y:ys)
  | otherwise = y: (insereOrd x ys)
```

Ordenação por Inserção em Haskell

Execução de um exemplo usando a função *insercao2*:

```
foldr f z []     = z
foldr f z (x:xs) = f x (foldr f z xs)
```

```
>insercao2 [3,1,5,2]

foldr insereOrd [] [3,1,5,2]
insereOrd 3 (foldr insereOrd [] [1,5,2])
      insereOrd 1 (foldr insereOrd [] [5,2])
            insereOrd 5 (foldr insereOrd [] [2])
                  insereOrd 2 (foldr insereOrd [] [])
                  insereOrd 2 []
            insereOrd 5 [2]
      insereOrd 1 [2,5]
insereOrd 3 [1,2,5]
[1,2,3,5]
```

Ordenação Rápida

Estratégia de ordenação:

- Considere o primeiro elemento da lista como um elemento *pivô* da ordenação.
- Particione a lista inicial em duas novas listas: a primeira deve conter os elementos menores que o elemento pivô, e a segunda contendo os maiores.
- Concatene a lista com os menores, seguida do elemento pivô, seguida dos elementos maiores.
- Repetir as ações acima até que toda a lista esteja particionada.

Ordenação Rápida

Exemplo:

```
Lista Inicial: [3,7,2,6,1,4]

Inserção Ordenada
Lista Menores  -- Pivô -- Lista Maiores
   [2,1]       --  3   --    [7,6,4]
     |             |           |
     |             |           |
Particiona                 Particiona
Sub-lista          3       Sub-lista
  [2,1]                      [7,6,4]
     |
     |
LMen  P  LMai
[1]   2   []
     ...
```

Ordenação Rápida em Haskell

```
quicksort::(Ord a) => [a] -> [a]
quicksort [] = []
quicksort (s:xs) = quicksort [x|x <- xs,x < s]
                   ++ [s] ++
                   quicksort [x|x <- xs,x >= s]
```

Ordenação por Bolha

Estratégia de ordenação:

- São feitas trocas sucessivas entre elementos consecutivos: Se o elemento na posição *n* for maior que o da posição *n+1*, então trocamos os dois elementos de posição.
- Efetuar o procedimento de troca acima *i-1* vezes, sendo *i* o tamanho da lista.

Ordenação por Bolha

Exemplo:

```
Lista Inicial: [3,7,2,6,1,4]

1ª Troca: [3,2,6,1,4,7]
2ª Troca: [2,3,1,4,6,7]
3ª Troca: [2,1,3,4,6,7]
4ª Troca: [1,2,3,4,6,7]
5ª Troca: [1,2,3,4,6,7]
```

Ordenação por Bolha em Haskell

```
bolha [] = []
bolha lista = bolhaOrd lista (length lista)

bolhaOrd lista 0 = lista
bolhaOrd lista n = bolhaOrd (troca lista) (n-1)

troca [x] = [x]
troca (x:y:zs)
        | x > y     = y : troca (x:zs)
        | otherwise = x : troca (y:zs)
```

Bibliografia

1. *Haskell - Uma abordagem prática*. Cláudio César de Sá e Márcio Ferreira da Silva. Novatec, 2006.
2. *Haskell - The craft of functional programming*. Simon Thompson. Pearson, Addison-Wesley, 1999.