

Strings

Busca em Strings

Prof. Edson Alves - UnB/FGA

2019

1. Busca em Strings
2. Busca Completa

Busca em Strings

Definição de busca em strings

- A busca é o algoritmo fundamental dentre os algoritmos de strings
- Ela equivalente, em importância, aos algoritmos de ordenação no estudo de algoritmos
- A busca em strings consiste em determinar se uma string P , de tamanho m , ocorre ou não em uma string S , de tamanho n
- Uma variante comum é determinar o número de ocorrências de P em S

Algoritmos de busca em strings

- Os principais algoritmos de busca em strings são
 1. a busca completa
 2. o algoritmo de Rabin-Karp
 3. o algoritmo de Knuth-Morris-Pratt
 4. a função z
 5. o algoritmo de Boyer-Moore
- O primeiro deles é de fácil entendimento e codificação
- Os demais são conceitualmente mais sofisticados, e podem compreender duas etapas: pré-processamento e busca
- Esta sofisticação dificulta a implementação, mas traz ganhos na complexidade assintótica da busca em relação à busca completa

Busca Completa

Busca completa em strings

- A busca completa compara cada uma das $n - m + 1$ substrings de tamanho m de S com P , reportando cada igualdade
- Como a comparação tem complexidade $O(m)$ e o número de substrings é $O(n)$, o algoritmo tem complexidade $O(mn)$ no pior caso
- É preciso ter cuidado com os limites do laço, a depender da representação de strings utilizada, para que todas as substrings sejam verificadas

Pseudocódigo para a busca completa em strings

Algoritmo 1 Busca completa em Strings

Require: Duas strings P e S

Ensure: O número de ocorrências occ de P em S

```
1: function SEARCH( $P, S$ )  
2:    $m \leftarrow |P|$   
3:    $n \leftarrow |S|$   
4:    $occ \leftarrow 0$   
5:    $i \leftarrow 1$   
6:   while  $|S[i..n]| \leq m$  do  
7:     if  $S[i..(i + m - 1)] = P$  then  
8:        $occ \leftarrow occ + 1$   
9:        $i \leftarrow i + 1$   
10:  return  $occ$ 
```

Implementação da busca completa em Haskell

```
1 occurrences :: String -> String -> Int
2 occurrences _ [] = 0
3 occurrences ps xs = c + occurrences ps (tail xs) where
4     m = length ps
5     c = if take m xs == ps then 1 else 0
```

Implementação da busca completa em C++

```
1 int occurrences(const string& P, const string& S)
2 {
3     int m = P.size();
4     int n = S.size();
5     int occ = 0;
6
7     for (int i = 0; i <= n - m; ++i)
8         occ += (P == S.substr(i, m) ? 1 : 0);
9
10    return occ;
11 }
```

1. **CROCHEMORE**, Maxime; **RYTTER**, Wojciech. *Jewels of Stringology: Text Algorithms*, WSPC, 2002.
2. **HALIM**, Steve; **HALIM**, Felix. *Competitive Programming 3*, Lulu, 2013.