Strings

Definição de string

Prof. Edson Alves - UnB/FGA

2018

Sumário

1. Motivação
2. Definições
3. Strings Notáveis

Motivação

Motivações para o estudo de strings

- Sequências de caracteres, ou strings, constituem uma maneira natural de representar informações

Motivações para o estudo de strings

- Sequências de caracteres, ou strings, constituem uma maneira natural de representar informações
- As strings aparecem em diversas áreas além da Computação, como a Biologia (estudo das moléculas e DNA), Letras (ortografia, sintaxe e morfologia), Criptografia (codificação e decodificação de mensagens), dentre outras

Motivações para o estudo de strings

- Sequências de caracteres, ou strings, constituem uma maneira natural de representar informações
- As strings aparecem em diversas áreas além da Computação, como a Biologia (estudo das moléculas e DNA), Letras (ortografia, sintaxe e morfologia), Criptografia (codificação e decodificação de mensagens), dentre outras
- O algoritmo fundamental para o estudo e entendimento de strings é o *pattern matching*, que consiste na localização informações (padrões) em um texto (string)

Motivações para o estudo de strings

- Sequências de caracteres, ou strings, constituem uma maneira natural de representar informações
- As strings aparecem em diversas áreas além da Computação, como a Biologia (estudo das moléculas e DNA), Letras (ortografia, sintaxe e morfologia), Criptografia (codificação e decodificação de mensagens), dentre outras
- O algoritmo fundamental para o estudo e entendimento de strings é o *pattern matching*, que consiste na localização informações (padrões) em um texto (string)
- A importância do *pattern matching* para o estudo das strings equivale à importância dos algoritmos de ordenação no estudo de algoritmos

Motivações para o estudo de strings

- Os padrões a serem localizados podem ser exatos, ou escritos em uma representação que utiliza caracteres especiais para marcar sequências ou repetições, denominada *regex* (*regular expressions*)

Motivações para o estudo de strings

- Os padrões a serem localizados podem ser exatos, ou escritos em uma representação que utiliza caracteres especiais para marcar sequências ou repetições, denominada *regex* (*regular expressions*)
- A linguagem awk (Aho, Weinberger, Kernighan) é interamente baseada em expressões regulares e é focada na manipulação de strings

Motivações para o estudo de strings

- Os padrões a serem localizados podem ser exatos, ou escritos em uma representação que utiliza caracteres especiais para marcar sequências ou repetições, denominada *regex* (*regular expressions*)
- A linguagem awk (Aho, Weinberger, Kernighan) é interamente baseada em expressões regulares e é focada na manipulação de strings
- O ambiente UNIX dispõe de várias ferramentas para textos (`grep`, `cat`, `more`, `less` `sed`, `diff`, etc), que permitem *pattern matching*, exibição, busca, identificação, filtragem, e manipulação de strings, dentre outros

Motivações para o estudo de strings

- Os padrões a serem localizados podem ser exatos, ou escritos em uma representação que utiliza caracteres especiais para marcar sequências ou repetições, denominada *regex* (*regular expressions*)
- A linguagem awk (Aho, Weinberger, Kernighan) é interamente baseada em expressões regulares e é focada na manipulação de strings
- O ambiente UNIX dispõe de várias ferramentas para textos (`grep`, `cat`, `more`, `less` `sed`, `diff`, etc), que permitem *pattern matching*, exibição, busca, identificação, filtragem, e manipulação de strings, dentre outros
- Estas ferramentas podem ser utilizadas isoladamente ou em conjunto, oferecendo uma grande gama de opções aos seus usuários

Definições

Definição de string

- Um alfabeto A é um conjunto finito de símbolos

Definição de string

- Um alfabeto A é um conjunto finito de símbolos
- Os elementos de um alfabeto A são denominados letras, caracteres ou símbolos

Definição de string

- Um alfabeto A é um conjunto finito de símbolos
- Os elementos de um alfabeto A são denominados letras, caracteres ou símbolos
- Exemplos de alfabetos comumente utilizados são as letras maiúsculas e minúsculas, os dígitos decimais e a tabela ASCII

Definição de string

- Um alfabeto A é um conjunto finito de símbolos
- Os elementos de um alfabeto A são denominados letras, caracteres ou símbolos
- Exemplos de alfabetos comumente utilizados são as letras maiúsculas e minúsculas, os dígitos decimais e a tabela ASCII
- Uma string s (ou texto ou palavra) é uma sequência ordenada $s = \{a_1, a_2, \ldots, a_N\}$ de caracteres a_i de um alfabeto A

Definição de string

- Um alfabeto A é um conjunto finito de símbolos
- Os elementos de um alfabeto A são denominados letras, caracteres ou símbolos
- Exemplos de alfabetos comumente utilizados são as letras maiúsculas e minúsculas, os dígitos decimais e a tabela ASCII
- Uma string s (ou texto ou palavra) é uma sequência ordenada $s = \{a_1, a_2, \ldots, a_N\}$ de caracteres a_i de um alfabeto A
- O i-ésimo termo de s também é denotado por s_i ou $s[i]$

Definição de string

- Um alfabeto A é um conjunto finito de símbolos
- Os elementos de um alfabeto A são denominados letras, caracteres ou símbolos
- Exemplos de alfabetos comumente utilizados são as letras maiúsculas e minúsculas, os dígitos decimais e a tabela ASCII
- Uma string s (ou texto ou palavra) é uma sequência ordenada $s = \{a_1, a_2, \ldots, a_N\}$ de caracteres a_i de um alfabeto A
- O i-ésimo termo de s também é denotado por s_i ou $s[i]$
- O número N de elementos da sequência s pode ser notado como $|s|$

Substrings

- Um intervalo é uma subsequência contígua

$$s[i..j] = s[i]s[i+1]\dots s[j]$$

de elementos de s

Substrings

- Um intervalo é uma subsequência contígua

$$s[i..j] = s[i]s[i+1]\dots s[j]$$

  de elementos de s
- Observe que $|s[i..j]| = j - i + 1$

Substrings

- Um intervalo é uma subsequência contígua

$$s[i..j] = s[i]s[i+1]\dots s[j]$$

  de elementos de s
- Observe que $|s[i..j]| = j - i + 1$
- Uma substring b de s, com $|b| = M$, é uma string b tal que $b = s[(i+1)..(i+M)]$ para algum inteiro i

Substrings

- Um intervalo é uma subsequência contígua

$$s[i..j] = s[i]s[i+1]\ldots s[j]$$

  de elementos de s
- Observe que $|s[i..j]| = j - i + 1$
- Uma substring b de s, com $|b| = M$, é uma string b tal que $b = s[(i+1)..(i+M)]$ para algum inteiro i
- Uma subsequência $a = s[i_1]s[i_2]\ldots s[i_M]$ de uma string s pode ser obtida a partir da remoção de zero ou mais elementos de s, não necessariamente consecutivos

Substrings

- Um intervalo é uma subsequência contígua

$$s[i..j] = s[i]s[i+1]\ldots s[j]$$

  de elementos de s
- Observe que $|s[i..j]| = j - i + 1$
- Uma substring b de s, com $|b| = M$, é uma string b tal que $b = s[(i+1)..(i+M)]$ para algum inteiro i
- Uma subsequência $a = s[i_1]s[i_2]\ldots s[i_M]$ de uma string s pode ser obtida a partir da remoção de zero ou mais elementos de s, não necessariamente consecutivos
- Os inteiros $i_1, i_2, ..., i_M$ formam uma sequência crescente de índices de s (isto é, $i_u < i_v$ para $u < v$)

Prefixos e sufixos

- Um prefixo de uma string s é uma substring x, de tamanho $|x| = M$, tal que $x = s[1..M]$

Prefixos e sufixos

- Um prefixo de uma string s é uma substring x, de tamanho $|x| = M$, tal que $x = s[1..M]$
- Um sufixo y de s, de tamanho $|y| = T$, é uma substring de s tal que $y = s[(N - T + 1)..N]$, onde $|s| = N$

Prefixos e sufixos

- Um prefixo de uma string s é uma substring x, de tamanho $|x| = M$, tal que $x = s[1..M]$
- Um sufixo y de s, de tamanho $|y| = T$, é uma substring de s tal que $y = s[(N - T + 1)..N]$, onde $|s| = N$
- Uma borda B de uma string s é uma substring que é, simultaneamente, prefixo e sufixo de s

Prefixos e sufixos

- Um prefixo de uma string s é uma substring x, de tamanho $|x| = M$, tal que $x = s[1..M]$
- Um sufixo y de s, de tamanho $|y| = T$, é uma substring de s tal que $y = s[(N - T + 1)..N]$, onde $|s| = N$
- Uma borda B de uma string s é uma substring que é, simultaneamente, prefixo e sufixo de s
- Uma vez que a string vazia (isto é, $|s| = 0$) e a própria string s são sempre bordas (triviais) de s, define-se $border(s)$ como a mais longa (de maior tamanho) dentre as bordas de s que são distintas da própria string s

Prefixos e sufixos

- Um prefixo de uma string s é uma substring x, de tamanho $|x| = M$, tal que $x = s[1..M]$
- Um sufixo y de s, de tamanho $|y| = T$, é uma substring de s tal que $y = s[(N - T + 1)..N]$, onde $|s| = N$
- Uma borda B de uma string s é uma substring que é, simultaneamente, prefixo e sufixo de s
- Uma vez que a string vazia (isto é, $|s| = 0$) e a própria string s são sempre bordas (triviais) de s, define-se $border(s)$ como a mais longa (de maior tamanho) dentre as bordas de s que são distintas da própria string s
- Por exemplo, as strings `"ame"`, `"rica"` e `"a"` são exemplos de prefixo, sufixo e borda da string `"america"`, respectivamente

Período de uma string

- Um período de uma string s é um inteiro p, $0 < p \leq |s|$ tal que $s[i] = s[i+p]$, para todo $i = 0, 1, \ldots, |s| - p$

Período de uma string

- Um período de uma string s é um inteiro p, $0 < p \leq |s|$ tal que $s[i] = s[i + p]$, para todo $i = 0, 1, \ldots, |s| - p$
- Para qualquer string, $|s|$ é um período, de modo que define-se $period(s)$ como o menor período de s

Período de uma string

- Um período de uma string s é um inteiro p, $0 < p \leq |s|$ tal que $s[i] = s[i+p]$, para todo $i = 0, 1, \ldots, |s| - p$
- Para qualquer string, $|s|$ é um período, de modo que define-se $period(s)$ como o menor período de s
- A string s é dita periódica se $period(s) \leq |s|/2$

Período de uma string

- Um período de uma string s é um inteiro p, $0 < p \leq |s|$ tal que $s[i] = s[i+p]$, para todo $i = 0, 1, \ldots, |s| - p$
- Para qualquer string, $|s|$ é um período, de modo que define-se $period(s)$ como o menor período de s
- A string s é dita periódica se $period(s) \leq |s|/2$
- Por exemplo, para as strings $s_1 =$ `"marítima"`, $s_2 =$ `"ticotico"` e $s_3 =$ `"Brasilia"`, temos $period(s_1) = 6, period(s_2) = 4$ e $period(s_3) = 8$

Período de uma string

- Um período de uma string s é um inteiro p, $0 < p \leq |s|$ tal que $s[i] = s[i+p]$, para todo $i = 0, 1, \ldots, |s| - p$
- Para qualquer string, $|s|$ é um período, de modo que define-se $period(s)$ como o menor período de s
- A string s é dita periódica se $period(s) \leq |s|/2$
- Por exemplo, para as strings $s_1 =$ `"marítima"`, $s_2 =$ `"ticotico"` e $s_3 =$ `"Brasilia"`, temos $period(s_1) = 6, period(s_2) = 4$ e $period(s_3) = 8$
- Dentre as três, apenas s_2 é períodica

Lemas de Periodicidade

Os diferentes períodos de uma mesma string s se relacionam de uma maneira não trivial, que pode ser expressa pelos dois lemas a seguir.

Lemas de Periodicidade

Os diferentes períodos de uma mesma string s se relacionam de uma maneira não trivial, que pode ser expressa pelos dois lemas a seguir.

Lema da Periodicidade

Sejam p e q dois períodos de uma string s. Se $p + q < |s|$, então $\mathrm{mdc}(p, q)$ também é período de s.

Lemas de Periodicidade

Os diferentes períodos de uma mesma string s se relacionam de uma maneira não trivial, que pode ser expressa pelos dois lemas a seguir.

Lema da Periodicidade

Sejam p e q dois períodos de uma string s. Se $p+q<|s|$, então $\mathrm{mdc}(p,q)$ também é período de s.

Lema Forte da Periodicidade

Sejam p e q dois períodos de uma string s. Se $p+q-\mathrm{mdc}(p,q)\leq|s|$, então mdc$(p,q)$ também é período de s.

Relação entre períodos e bordas

- Há uma interessante relação entre bordas e períodos

Relação entre períodos e bordas

- Há uma interessante relação entre bordas e períodos
- A sequência

$$|s| - |border(s)|, |s| - |border^2(s)|, ..., |s| - |border^k(s)|$$

  é a sequência crescente de todos os possíveis períodos de s, onde k é o menor inteiro positivo tal que $border^k(s)$ é uma string vazia

Relação entre períodos e bordas

- Há uma interessante relação entre bordas e períodos
- A sequência

$$|s| - |border(s)|, |s| - |border^2(s)|, ..., |s| - |border^k(s)|$$

  é a sequência crescente de todos os possíveis períodos de s, onde k é o menor inteiro positivo tal que $border^k(s)$ é uma string vazia
- Por exemplo, para a string $s =$ `"teteatete"`, tem-se $|s| = 9$ e

$$\begin{aligned} border(s) &= \texttt{"tete"} \\ border^2(s) &= border(\texttt{"tete"}) = \texttt{"te"} \\ border^3(s) &= border(\texttt{"te"}) = \texttt{""} \end{aligned}$$

  os quais formam a sequência de períodos $9 - 4 = 5, 9 - 2 = 7$ e $9 - 0 = 9$

Pattern matching

- O problema fundamental de strings é o *pattern matching*

Pattern matching

- O problema fundamental de strings é o *pattern matching*
- Dada uma string P, que representa um padrão, o *pattern matching* consiste em determinar se P ocorre ou não em s

Pattern matching

- O problema fundamental de strings é o *pattern matching*
- Dada uma string P, que representa um padrão, o *pattern matching* consiste em determinar se P ocorre ou não em s
- O *pattern matching* é um problema de decisão, isto é, a resposta é booleana: o padrão ocorre ou não, embora uma variante comum é determinar o índice da primeira posição onde P ocorre ou um valor sentinela, caso não ocorra

Pattern matching

- O problema fundamental de strings é o *pattern matching*
- Dada uma string P, que representa um padrão, o *pattern matching* consiste em determinar se P ocorre ou não em s
- O *pattern matching* é um problema de decisão, isto é, a resposta é booleana: o padrão ocorre ou não, embora uma variante comum é determinar o índice da primeira posição onde P ocorre ou um valor sentinela, caso não ocorra
- Em geral, $|P| \leq |s|$

Pattern matching

- O problema fundamental de strings é o *pattern matching*
- Dada uma string P, que representa um padrão, o *pattern matching* consiste em determinar se P ocorre ou não em s
- O *pattern matching* é um problema de decisão, isto é, a resposta é booleana: o padrão ocorre ou não, embora uma variante comum é determinar o índice da primeira posição onde P ocorre ou um valor sentinela, caso não ocorra
- Em geral, $|P| \leq |s|$
- Uma notação possível é $match(P, s)$

Strings Notáveis

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$
\begin{aligned}
F_0 &= \text{""} \\
F_1 &= \text{"b"} \\
F_2 &= \text{"a"} \\
F_n &= F_{(n-1)}F_{(n-2)}, n > 2
\end{aligned}
$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$F_0 = \text{""}$$
$$F_1 = \text{"b"}$$
$$F_2 = \text{"a"}$$
$$F_n = F_{(n-1)}F_{(n-2)}, n > 2$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci
- Por exemplo, $F_3 =$ "ab", $F_4 =$ "aba" e $F_5 =$ "abaab"

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$F_0 = ""$$

$$F_1 = "\text{b}"$$

$$F_2 = "\text{a}"$$

$$F_n = F_{(n-1)}F_{(n-2)}, n > 2$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci
- Por exemplo, $F_3 =$ "ab", $F_4 =$ "aba" e $F_5 =$ "abaab"
- Há 3 fatos notáveis a respeito das strings de Fibonacci:

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$
\begin{aligned}
F_0 &= \text{""} \\
F_1 &= \text{"b"} \\
F_2 &= \text{"a"} \\
F_n &= F_{(n-1)}F_{(n-2)}, n > 2
\end{aligned}
$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci
- Por exemplo, $F_3 =$ "ab", $F_4 =$ "aba" e $F_5 =$ "abaab"
- Há 3 fatos notáveis a respeito das strings de Fibonacci:
  1. removidas as duas últimas letras de uma string de Fibonacci, o resultado é um palíndromo

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$F_0 = ""$$
$$F_1 = "b"$$
$$F_2 = "a"$$
$$F_n = F_{(n-1)}F_{(n-2)}, n > 2$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci
- Por exemplo, $F_3 =$ "ab", $F_4 =$ "aba" e $F_5 =$ "abaab"
- Há 3 fatos notáveis a respeito das strings de Fibonacci:
  1. removidas as duas últimas letras de uma string de Fibonacci, o resultado é um palíndromo
  2. qualquer string de Fibonacci F_n com $n \geq 2$ é prefixo de outra string de Fibonacci

Strings de Fibonacci

- As strings de Fibonacci F_n, com $n \leq 0$, são definidas como

$$F_0 = ""$$

$$F_1 = "\text{b}"$$

$$F_2 = "\text{a}"$$

$$F_n = F_{(n-1)}F_{(n-2)}, n > 2$$

  onde a expressão $F_{(n-1)}F_{(n-2)}$ significa a concatenação das últimas duas strings de Fibonacci

- Por exemplo, $F_3 =$ "ab", $F_4 =$ "aba" e $F_5 =$ "abaab"
- Há 3 fatos notáveis a respeito das strings de Fibonacci:
  1. removidas as duas últimas letras de uma string de Fibonacci, o resultado é um palíndromo
  2. qualquer string de Fibonacci F_n com $n \geq 2$ é prefixo de outra string de Fibonacci
  3. todas strings de Fibonacci F_n com $n \geq 2$ são prefixos de F_∞

Prefixos de Thue-Morse

- Considere a string infinita T_∞, definida da seguinte maneira, onde $g(k)$ é o número dígitos 1 (um) na representação binária do inteiro não-negativo k:

$$T_\infty(k) = \begin{cases} \texttt{'a'}, & \text{se } g(k) \text{ é par} \\ \texttt{'b'}, & \text{caso contrário} \end{cases}$$

Prefixos de Thue-Morse

- Considere a string infinita T_∞, definida da seguinte maneira, onde $g(k)$ é o número dígitos 1 (um) na representação binária do inteiro não-negativo k:

$$T_\infty(k) = \begin{cases} \texttt{'a'}, & \text{se } g(k) \text{ é par} \\ \texttt{'b'}, & \text{caso contrário} \end{cases}$$

- Os prefixos de Thue-Morse $T(n)$ são os prefixos de T_∞ de tamanho 2^n

Prefixos de Thue-Morse

- Considere a string infinita T_∞, definida da seguinte maneira, onde $g(k)$ é o número dígitos 1 (um) na representação binária do inteiro não-negativo k:

$$T_\infty(k) = \begin{cases} \texttt{'a'}, & \text{se } g(k) \text{ é par} \\ \texttt{'b'}, & \text{caso contrário} \end{cases}$$

- Os prefixos de Thue-Morse $T(n)$ são os prefixos de T_∞ de tamanho 2^n
- Os quatro primeiros prefixos de Thue-Morse são: $T(1) =$ "ab", $T(2) =$ "abba", $T(3) =$ "abbabaab" e $T(4) =$ "abbabaabbaababba"

Prefixos de Thue-Morse

- Considere a string infinita T_∞, definida da seguinte maneira, onde $g(k)$ é o número dígitos 1 (um) na representação binária do inteiro não-negativo k:

$$T_\infty(k) = \begin{cases} \texttt{'a'}, & \text{se } g(k) \text{ é par} \\ \texttt{'b'}, & \text{caso contrário} \end{cases}$$

- Os prefixos de Thue-Morse $T(n)$ são os prefixos de T_∞ de tamanho 2^n
- Os quatro primeiros prefixos de Thue-Morse são: $T(1) =$ "ab", $T(2) =$ "abba", $T(3) =$ "abbabaab" e $T(4) =$ "abbabaabbaababba"
- Estas strings são livres de *overlaps*, isto é, não existe nenhuma string não vazia s que ocorre em duas posições distintas de $T(n)$ com distância entre estas posições menor do que $|s|$

Prefixos de Thue-Morse

- Considere a string infinita T_∞, definida da seguinte maneira, onde $g(k)$ é o número dígitos 1 (um) na representação binária do inteiro não-negativo k:

$$T_\infty(k) = \begin{cases} \texttt{'a'}, & \text{se } g(k) \text{ é par} \\ \texttt{'b'}, & \text{caso contrário} \end{cases}$$

- Os prefixos de Thue-Morse $T(n)$ são os prefixos de T_∞ de tamanho 2^n
- Os quatro primeiros prefixos de Thue-Morse são: $T(1) =$ "ab", $T(2) =$ "abba", $T(3) =$ "abbabaab" e $T(4) =$ "abbabaabbaababba"
- Estas strings são livres de *overlaps*, isto é, não existe nenhuma string não vazia s que ocorre em duas posições distintas de $T(n)$ com distância entre estas posições menor do que $|s|$
- Também são livre de quadrados: não existe um string s tal que a concatenação de s consigo mesma seja substring de $T(n)$

Palavras binárias P_n

- A palavra binária P_n é obtida a partir da n-ésima linha do triângulo de Pascal, onde seu i-ésimo caractere é dado por

$$P_n[i] = \binom{n}{i} \pmod 2$$

Palavras binárias P_n

- A palavra binária P_n é obtida a partir da n-ésima linha do triângulo de Pascal, onde seu i-ésimo caractere é dado por

$$P_n[i] = \binom{n}{i} \pmod 2$$

- As primeias 5 palavras binárias P_n são

$$P_0 = \texttt{"1"}$$
$$P_1 = \texttt{"11"}$$
$$P_2 = \texttt{"101"}$$
$$P_3 = \texttt{"1111"}$$
$$P_4 = \texttt{"10001"}$$

Palavras binárias P_n

- A palavra binária P_n é obtida a partir da n-ésima linha do triângulo de Pascal, onde seu i-ésimo caractere é dado por

$$P_n[i] = \binom{n}{i} \pmod 2$$

- As primeias 5 palavras binárias P_n são

$$P_0 = \texttt{"1"}$$
$$P_1 = \texttt{"11"}$$
$$P_2 = \texttt{"101"}$$
$$P_3 = \texttt{"1111"}$$
$$P_4 = \texttt{"10001"}$$

- O número de ocorrências do caractere `'1'` em P_n é igual a $2^{g(n)}$, onde $g(k)$ tem a mesma definição dada nos prefixos de Thue-Morse

String de dígitos

- Considere a string infinita W composta pela representação decimal dos números naturais consecutivos, isto é,

$$W = \texttt{"0123456789101112131415161718192021222324 25..."}$$

String de dígitos

- Considere a string infinita W composta pela representação decimal dos números naturais consecutivos, isto é,

$$W = \texttt{"0123456789101112131415161718192021222232425..."}$$

- W_n é o prefixo de W de tamanho n

String de dígitos

- Considere a string infinita W composta pela representação decimal dos números naturais consecutivos, isto é,

$$W = \texttt{"0123456789101112131415161718192021222324 25..."}$$

- W_n é o prefixo de W de tamanho n
- Seja s uma string composta por dígitos decimais. A função $occ_n(s)$ computa o número de ocorrências de s em W_n

String de dígitos

- Considere a string infinita W composta pela representação decimal dos números naturais consecutivos, isto é,

$$W = \text{"0123456789101112131415161718192021222324 25..."}$$

- W_n é o prefixo de W de tamanho n
- Seja s uma string composta por dígitos decimais. A função $occ_n(s)$ computa o número de ocorrências de s em W_n
- Para duas strings s, t de mesmo tamanho,

$$\lim_{n \to \infty} \left(\frac{occ_n(s)}{occ_n(t)} \right) = 1$$

String de dígitos

- Considere a string infinita W composta pela representação decimal dos números naturais consecutivos, isto é,

$$W = \texttt{"0123456789101112131415161718192021222232425..."}$$

- W_n é o prefixo de W de tamanho n
- Seja s uma string composta por dígitos decimais. A função $occ_n(s)$ computa o número de ocorrências de s em W_n
- Para duas strings s, t de mesmo tamanho,

$$\lim_{n \to \infty} \left(\frac{occ_n(s)}{occ_n(t)} \right) = 1$$

- Esta propriedade dá um sentido "randômico" para a sequência W

Strings de Bruijin

- Seja $A = \{a, b\}$. Existem 2^k strings de tamanho k formadas por elementos de A

Strings de Bruijin

- Seja $A = \{a, b\}$. Existem 2^k strings de tamanho k formadas por elementos de A
- Uma pergunta natural que surge é: qual é o comprimento mínimo $\gamma(k)$ de uma string que contenha todas estas substrings?

Strings de Bruijin

- Seja $A = \{a, b\}$. Existem 2^k strings de tamanho k formadas por elementos de A
- Uma pergunta natural que surge é: qual é o comprimento mínimo $\gamma(k)$ de uma string que contenha todas estas substrings?
- Um limite inferior é $\gamma(k) = 2^k + k - 1$, pois qualquer string menor não teria 2^k substrings de tamanho k

Strings de Bruijin

- Seja $A = \{a, b\}$. Existem 2^k strings de tamanho k formadas por elementos de A
- Uma pergunta natural que surge é: qual é o comprimento mínimo $\gamma(k)$ de uma string que contenha todas estas substrings?
- Um limite inferior é $\gamma(k) = 2^k + k - 1$, pois qualquer string menor não teria 2^k substrings de tamanho k
- Efetivamente, $\gamma(k) = 2^k + k - 1$

Strings de Bruijin

- Seja $A = \{a, b\}$. Existem 2^k strings de tamanho k formadas por elementos de A
- Uma pergunta natural que surge é: qual é o comprimento mínimo $\gamma(k)$ de uma string que contenha todas estas substrings?
- Um limite inferior é $\gamma(k) = 2^k + k - 1$, pois qualquer string menor não teria 2^k substrings de tamanho k
- Efetivamente, $\gamma(k) = 2^k + k - 1$
- Uma string com este tamanho, contendo todas as substrings de tamanho k formadas por elementos de A, é denominada string de Bruijin

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler
- Seja G_k um grafo cujos vértices são todas as strings de elementos de A de tamanho $k - 1$

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler
- Seja G_k um grafo cujos vértices são todas as strings de elementos de A de tamanho $k-1$
- Para qualquer string $x = a_1 a_2 \dots a_{k-1}$ temos duas arestas direcionadas:

$$a_1 a_2 \dots a_{k-1} \xrightarrow{a} a_2 a_3 \dots a_{k-1} a$$

$$a_1 a_2 \dots a_{k-1} \xrightarrow{b} a_2 a_3 \dots a_{k-1} b$$

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler
- Seja G_k um grafo cujos vértices são todas as strings de elementos de A de tamanho $k-1$
- Para qualquer string $x = a_1a_2 \ldots a_{k-1}$ temos duas arestas direcionadas:

$$a_1a_2 \ldots a_{k-1} \xrightarrow{a} a_2a_3 \ldots a_{k-1}a$$
$$a_1a_2 \ldots a_{k-1} \xrightarrow{b} a_2a_3 \ldots a_{k-1}b$$

- Este grafo tem um ciclo de Euler direcionado, que contém cada aresta uma única vez

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler
- Seja G_k um grafo cujos vértices são todas as strings de elementos de A de tamanho $k-1$
- Para qualquer string $x = a_1a_2\ldots a_{k-1}$ temos duas arestas direcionadas:

$$a_1a_2\ldots a_{k-1} \xrightarrow{a} a_2a_3\ldots a_{k-1}a$$
$$a_1a_2\ldots a_{k-1} \xrightarrow{b} a_2a_3\ldots a_{k-1}b$$

- Este grafo tem um ciclo de Euler direcionado, que contém cada aresta uma única vez
- Seja $a_1a_2\ldots a_N$ a sequência de arestas do ciclo de Euler

Strings de Bruijin e Ciclos de Euler

- Há uma relação entre strings de Bruijin e ciclos de Euler
- Seja G_k um grafo cujos vértices são todas as strings de elementos de A de tamanho $k-1$
- Para qualquer string $x = a_1a_2\ldots a_{k-1}$ temos duas arestas direcionadas:

$$a_1a_2\ldots a_{k-1} \xrightarrow{a} a_2a_3\ldots a_{k-1}a$$
$$a_1a_2\ldots a_{k-1} \xrightarrow{b} a_2a_3\ldots a_{k-1}b$$

- Este grafo tem um ciclo de Euler direcionado, que contém cada aresta uma única vez
- Seja $a_1a_2\ldots a_N$ a sequência de arestas do ciclo de Euler
- Segue que $N = 2^k$, e que a sequência abaixo forma uma string de Bruijin:

$$a_1a_2\ldots a_Na_1a_2\ldots a_{k-1}$$

Referências

1. **HALIM**, Steve; **HALIM**, Felix. *Competitive Programming 3*, Lulu, 2013.
2. **CROCHEMORE**, Maxime; **RYTTER**, Wojciech. *Jewels of Stringology: Text Algorithms*, WSPC, 2002.
3. Wikipedia. ASCII, acesso em 06/02/2017.
4. Wikipedia. Fibonacci Word, acesso em 07/02/2017.