GSI024 - Organização e Recuperação de Informação
Dicionários e recuperação tolerante

Marcelo Keese Albertini

Conteúdo

Conteúdo

Distinção entre tipo e token

- Token – uma instância de uma palavra um termo em um documento
- Tipo – uma classe de equivalência de tokens
- *In June, the dog likes to chase the cat in the barn.*
- 12 tokens, 9 tipos

Problemas na tokenização

- O que são os delimitadores? Espaço? Apostófre? Hífen?
- Para cada um desses: às vezes delimitam, às vezes sim
- Não há espaços em vários idiomas! (e.g., Chinês)
- Não há espaços em subtantivos compostos em Holandês, Alemão, Sueco (*Lebensversicherungsgesellschaftsangestellter*)

Problemas com classe de equivalência

- Um termo uma classe de equivalência de tokens
- Como definir classes de equivalência?
- Números (3/20/91 vs. 20/3/91)
- Mudança de maiúscula/minúscula
- Stemming, Stemmer de Porter
- Análise morfológica: inflexão vs. derivação
- Problemas de classe de equivalência em outros idiomas
  - Morfologias mais complexas
  - Finlandês: um verbo pode tem 12 mil formas diferentes
  - Acentos, tremas

Índices posicionais

- Listas de índices não posicionais: cada referência é um simples docID
- Listas de índices posicionais: cada referência é um docID e uma lista de posições
- Exemplo de consulta: *"to*$_1$ *be*$_2$ *or*$_3$ *not*$_4$ *to*$_5$ *be*$_6$*"*

TO, 993427:
⟨ 1: ⟨7, 18, 33, 72, 86, 231⟩;
2: ⟨1, 17, 74, 222, 255⟩;
4: ⟨8, 16, 190, 429, 433⟩;
5: ⟨363, 367⟩;
7: ⟨13, 23, 191⟩; . . . ⟩

BE, 178239:
⟨ 1: ⟨17, 25⟩;
4: ⟨17, 191, 291, 430, 434⟩;
5: ⟨14, 19, 101⟩; . . . ⟩

Documento 4 é um resultado!

Índices posicionais

- Com um índice posicional, podemos responder consultas de expressões.
- Com um índice posicional, podemos responder consultas de proximidades.

Conteúdo

Conteúdo

- Recuperação tolerante: o que fazer se não há correspondência exata entre termos de consulta e termos de documentos

Conteúdo

- Recuperação tolerante: o que fazer se não há correspondência exata entre termos de consulta e termos de documentos
- Consultas com caracteres curingas

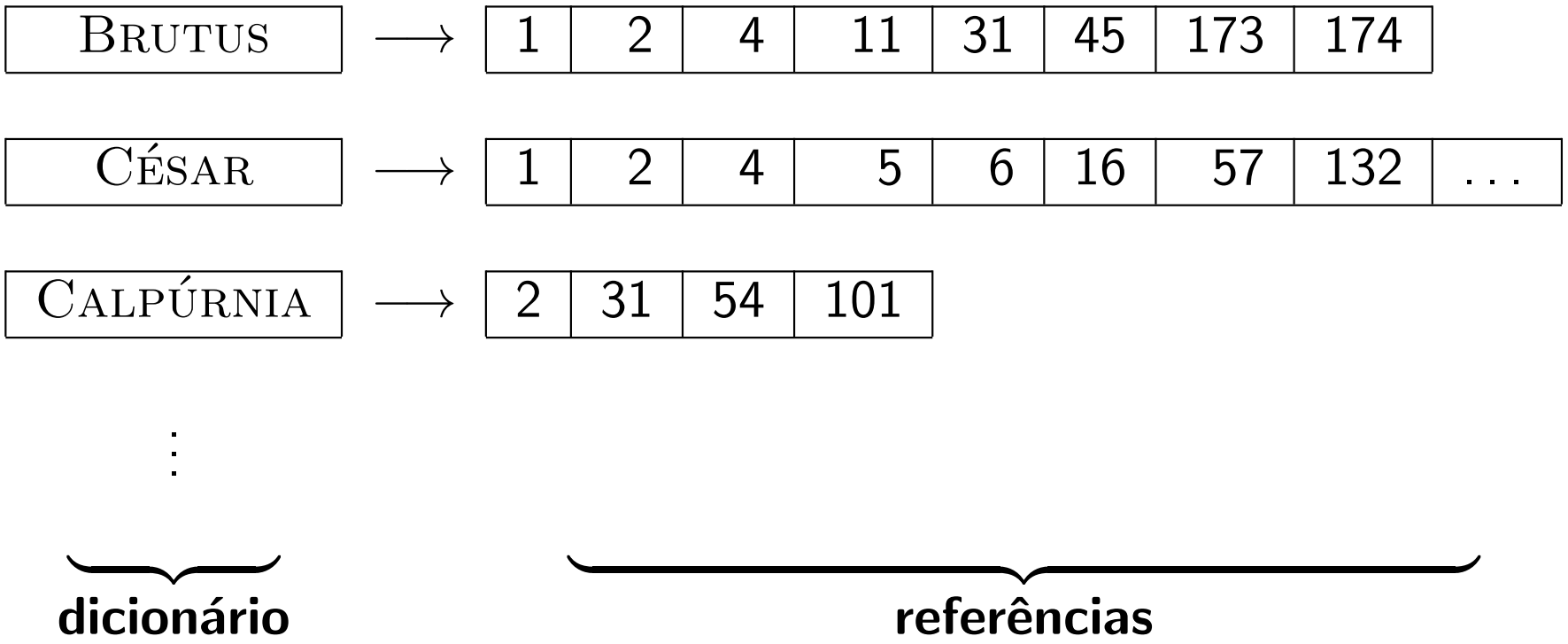
Conteúdo

- Recuperação tolerante: o que fazer se não há correspondência exata entre termos de consulta e termos de documentos
- Consultas com caracteres curingas
- Correção ortográfica

Conteúdo

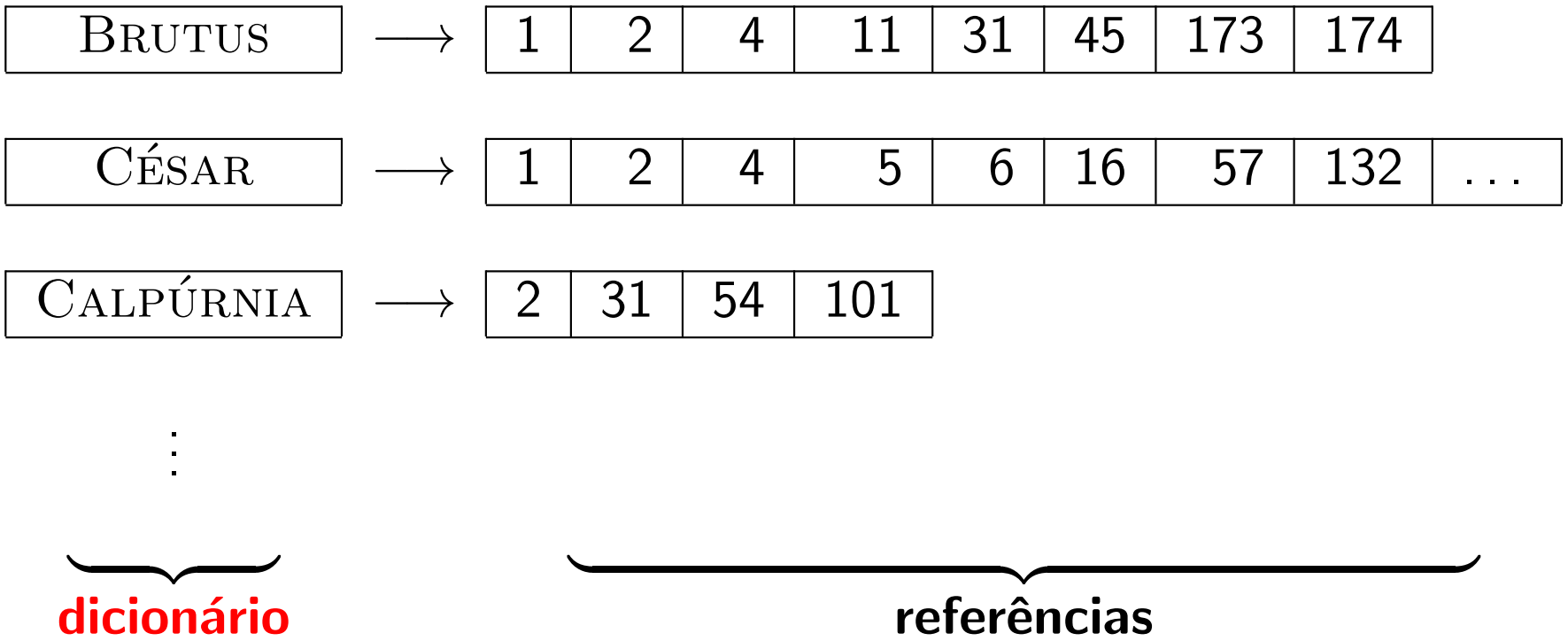
Índice invertido

Para cada termo t, armazenamos uma lista de todos os documentos que contém t.



Índice invertido

Para cada termo t, armazenamos uma lista de todos os documentos que contém t.



Dicionários

- Um dicionário é uma estrutura de dados para armazenar o vocabulário de termos

Dicionários

- Um dicionário é uma estrutura de dados para armazenar o vocabulário de termos
- Vocabulário: os dados

Dicionários

- Um dicionário é uma estrutura de dados para armazenar o vocabulário de termos
- Vocabulário: os dados
- Dicionário: a estrutura de dados para armazenar um vocabulário

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:
  - frequência em documentos

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:
  - frequência em documentos
  - ponteiro para lista de índices

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:
  - frequência em documentos
  - ponteiro para lista de índices
  - ...

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:
  - frequência em documentos
  - ponteiro para lista de índices
  - ...
- Assuma no momento que podemos armazenar essa informação em uma estrutura de tamanho fixo

Dicionário como um array de entradas de tamanho fixo

- Para cada termo, precisamos armazenar os seguintes itens:
  - frequência em documentos
  - ponteiro para lista de índices
  - ...
- Assuma no momento que podemos armazenar essa informação em uma estrutura de tamanho fixo
- Assuma que armazenamos essas estruturas em um array

dicionário como um array de entradas de tamanho fixo

termo	frequência no documento	ponteiro para lista de referências
a	656,265	$\longrightarrow$
aachen	65	$\longrightarrow$
...	...	...
zulu	221	$\longrightarrow$
espaço necessário:	20 bytes	4 bytes	4 bytes

Como procurar por um termo q_i nesse array durante o tempo da consulta? Isso é, qual estrutura de dados usamos para localizar a entrada (linha) em um array onde q_i é armazenado?

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores
- Alguns ORI usam hashes, outros, árvores

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores
- Alguns ORI usam hashes, outros, árvores
- Critério para quando usar hashes vs. árvores:

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores
- Alguns ORI usam hashes, outros, árvores
- Critério para quando usar hashes vs. árvores:
  - Há um número fixo de termo ou esses aumentarão?

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores
- Alguns ORI usam hashes, outros, árvores
- Critério para quando usar hashes vs. árvores:
  - Há um número fixo de termo ou esses aumentarão?
  - Quais são as frequências relativas com as quais várias chaves serão acessados?

Estrutura de dados para procurar por termo

- Duas classes principais de estruturas de dados: hashes e árvores
- Alguns ORI usam hashes, outros, árvores
- Critério para quando usar hashes vs. árvores:
  - Há um número fixo de termo ou esses aumentarão?
  - Quais são as frequências relativas com as quais várias chaves serão acessados?
  - Quantos termos são prováveis para ter?

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore
  - tempo de busca é constante

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore
  - tempo de busca é constante
- Contras

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore
  - tempo de busca é constante
- Contras
  - não permite encontrar variantes (*resume* vs. *résumé*)

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore
  - tempo de busca é constante
- Contras
  - não permite encontrar variantes (*resume* vs. *résumé*)
  - sem busca de prefixo (todos os termos iniciando com *automat*)

Hashes

- Cada termo no vocabulário é referenciado por uma chave-hash
- Tentar evitar colisões
- Durante a consulta, faça: obtenha chave hash do termo, trate colisões, localizar entrada no array de tamanho fixo
- Pros: buscar em uma hash é mais rápido que uma busca em uma árvore
  - tempo de busca é constante
- Contras
  - não permite encontrar variantes (*resume* vs. *résumé*)
  - sem busca de prefixo (todos os termos iniciando com *automat*)
  - necessário para reconstruir tabela hash periodicamente se vocabulário aumenta ao longo do tempo

Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).

Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária

Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária
- Busca é ligeiramente mais lento que com hash: $O(\log M)$, onde M é o tamanho do vocabulário

Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária
- Busca é ligeiramente mais lento que com hash: $O(\log M)$, onde M é o tamanho do vocabulário
- $O(\log M)$ somente vale para árvore balanceada

Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária
- Busca é ligeiramente mais lento que com hash: $O(\log M)$, onde M é o tamanho do vocabulário
- $O(\log M)$ somente vale para árvore balanceada
- Rebalanceamento de árvores binárias é caro

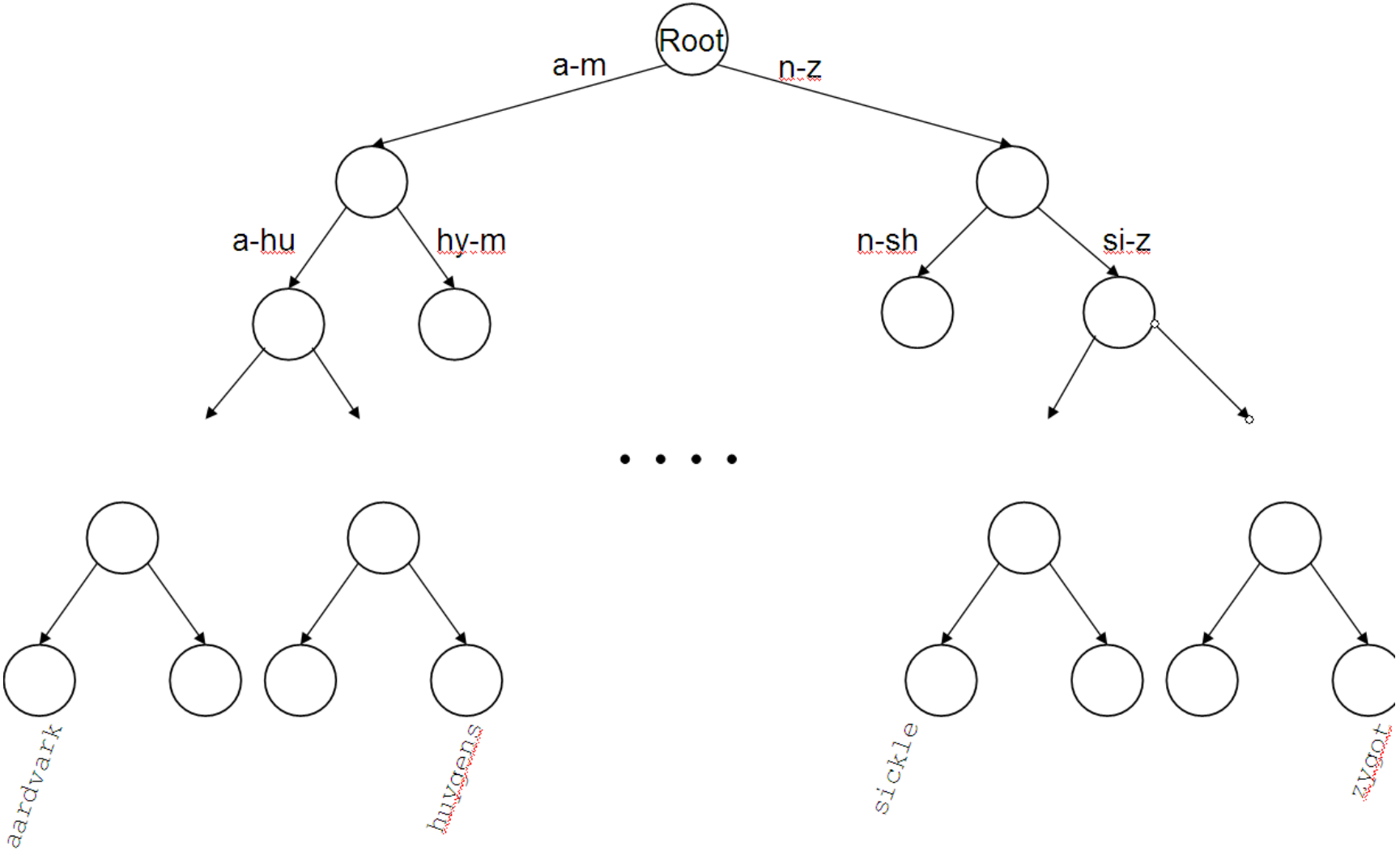
Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária
- Busca é ligeiramente mais lento que com hash: $O(\log M)$, onde M é o tamanho do vocabulário
- $O(\log M)$ somente vale para árvore balanceada
- Rebalanceamento de árvores binárias é caro
- Árvores B mitigam o problema de rebalanceamento

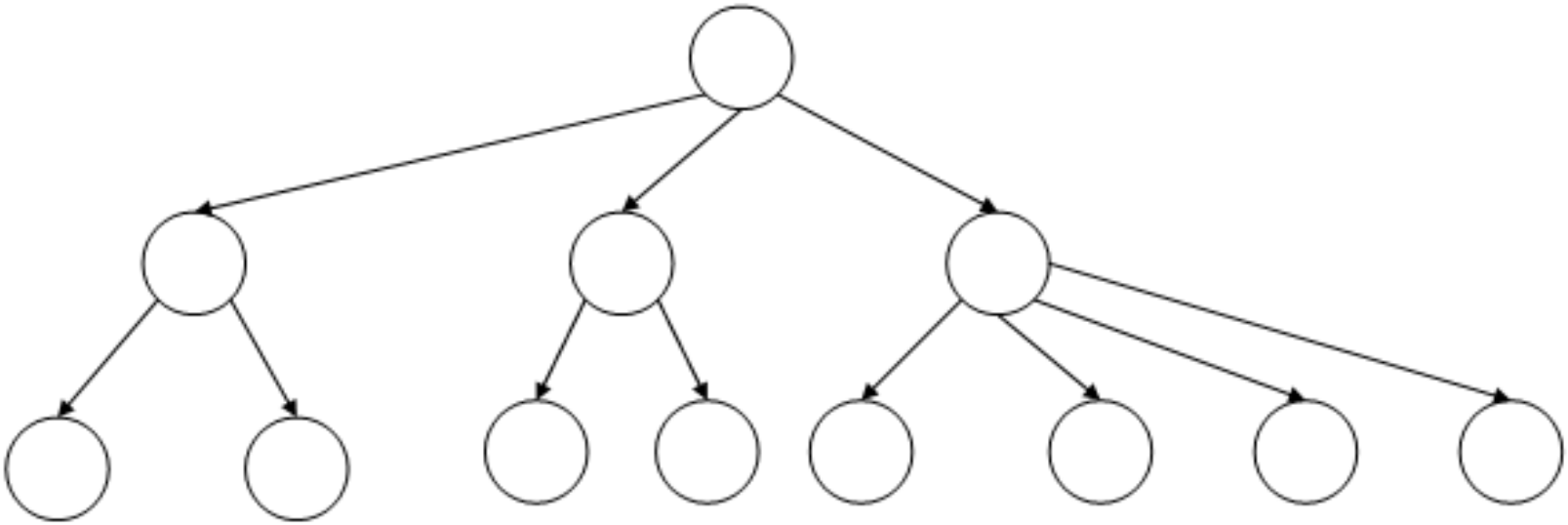
Árvores

- Árvores resolvem o problema com prefixos (encontre todos os termos iniciando com *automat*).
- Árvore mais simples: árvore binária
- Busca é ligeiramente mais lento que com hash: $O(\log M)$, onde M é o tamanho do vocabulário
- $O(\log M)$ somente vale para árvore balanceada
- Rebalanceamento de árvores binárias é caro
- Árvores B mitigam o problema de rebalanceamento
- Definição de árvore-B: todo nó interno tem um número de filhos no intervalo $[a, b]$ onde a, b são números inteiros positivos apropriados , e.g., $[2, 4]$.

Árvore binária



Árvore B



Conteúdo

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: $\text{mon} \leq t < \text{moo}$

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: mon $\leq t <$ moo
- *mon: encontrar todos documentos com termos terminando com *mon*

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: mon $\leq t <$ moo
- *mon: encontrar todos documentos com termos terminando com *mon*
  - Mantém uma árvore adicional para termos *ao contrário*

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: mon $\leq t <$ moo
- *mon: encontrar todos documentos com termos terminando com *mon*
  - Mantém uma árvore adicional para termos *ao contrário*
  - Então recupera todos termos t no intervalo: nom $\leq t <$ non

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: mon $\leq t <$ moo
- *mon: encontrar todos documentos com termos terminando com *mon*
  - Mantém uma árvore adicional para termos *ao contrário*
  - Então recupera todos termos t no intervalo: nom $\leq t <$ non
- Resultado: um conjunto de termos que correspondem a uma consulta com caracter curinga

Consultas com caracter curinga

- mon*: encontrar documentos com termos começando com *mon*
- Fácil com dicionário baseado em árvore B: recuperar todos termos t no intervalo: mon $\leq t <$ moo
- *mon: encontrar todos documentos com termos terminando com *mon*
  - Mantém uma árvore adicional para termos *ao contrário*
  - Então recupera todos termos t no intervalo: nom $\leq t <$ non
- Resultado: um conjunto de termos que correspondem a uma consulta com caracter curinga
- Então recupere documentos que contém quaisquer dos termos

Como manipular * no meio de um termo

- Exemplo: m*nchen

Como manipular * no meio de um termo

- Exemplo: m*nchen
- Podemos procurar m* e *nchen na árvore-B e interseccionar os dois conjuntos

Como manipular * no meio de um termo

- Exemplo: m*nchen
- Podemos procurar m* e *nchen na árvore-B e interseccionar os dois conjuntos
- Custoso

Como manipular * no meio de um termo

- Exemplo: m*nchen
- Podemos procurar m* e *nchen na árvore-B e interseccionar os dois conjuntos
- Custoso
- Alternativa: criar um índice permuterm

Como manipular * no meio de um termo

- Exemplo: m*nchen
- Podemos procurar m* e *nchen na árvore-B e interseccionar os dois conjuntos
- Custoso
- Alternativa: criar um índice permuterm
  - Ideia básica: rotacionar toda consulta com caracter curinga, tal que * ocorra no final

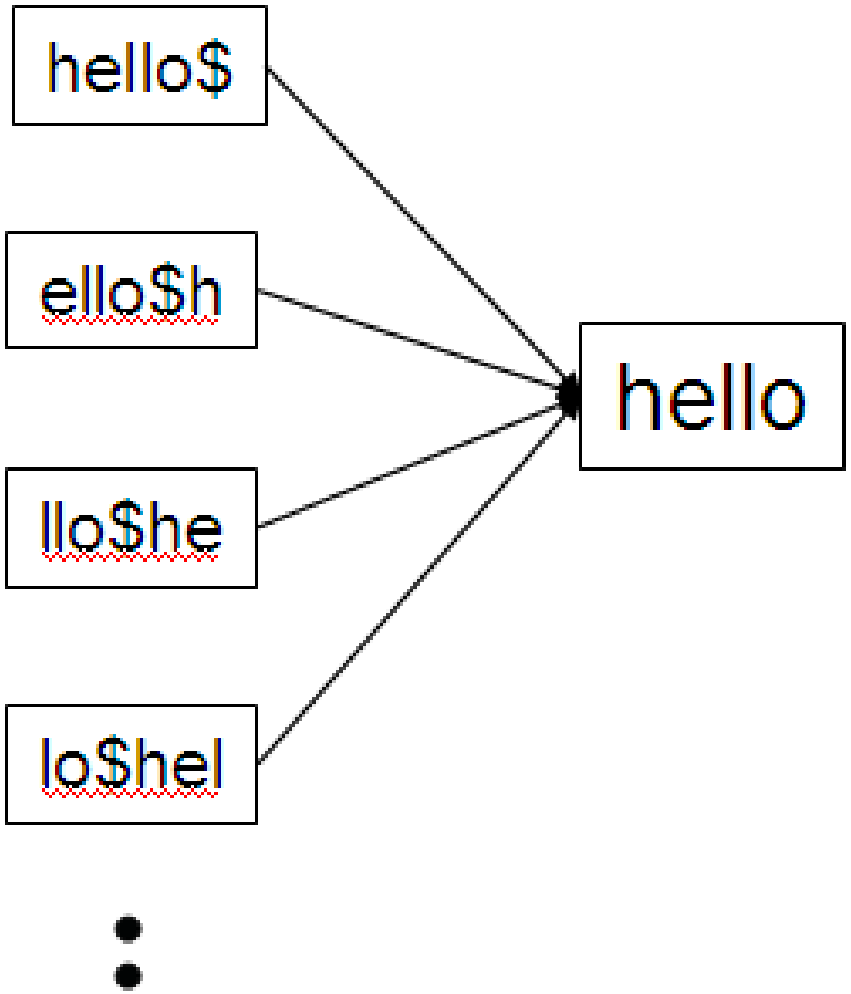
Como manipular * no meio de um termo

- Exemplo: m*nchen
- Podemos procurar m* e *nchen na árvore-B e interseccionar os dois conjuntos
- Custoso
- Alternativa: criar um índice permuterm
  - Ideia básica: rotacionar toda consulta com caracter curinga, tal que * ocorra no final
  - Guardar cada dessas rotações no dicionário, e.g., em uma árvore-B

Índice permuterm

- Para termo HELLO: adicionar *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell* para a árvore-B onde $ é um símbolo especial

Permuterm → termo mapeado



Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$
  - Para X*, busca por $X*

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$
  - Para X*, busca por $X*
  - Para *X, busca por X$*

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$
  - Para X*, busca por $X*
  - Para *X, busca por X$*
  - Para *X*, busca por X*

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$
  - Para X*, busca por $X*
  - Para *X, busca por X$*
  - Para *X*, busca por X*
  - Para X*Y, busca por Y$X*

Índice permuterm

- Para HELLO, armazenamos: *hello$*, *ello$h*, *llo$he*, *lo$hel*, e *o$hell*
- Consultas
  - Para X, busca por X$
  - Para X*, busca por $X*
  - Para *X, busca por X$*
  - Para *X*, busca por X*
  - Para X*Y, busca por Y$X*
  - Exemplo: para hel*o, busca o$hel*

Como processar uma busca no índice permuterm

- Passos
  - Rotacionar caractere curinga para a direita
  - Usar busca de prefixos em árvore de busca
- Problema: permuterm aumenta muito o tamanho do dicionário quando comparado a uma árvore B normal

Índices k-grama

- Mais eficiente em relação ao espaço que índice permuterm

Índices k-grama

- Mais eficiente em relação ao espaço que índice permuterm
- Enumerar todos os k-gramas (sequência de k caracteres) ocorrendo em um termo

Índices k-grama

- Mais eficiente em relação ao espaço que índice permuterm
- Enumerar todos os k-gramas (sequência de k caracteres) ocorrendo em um termo
- 2-gramas são chamados bigramas.

Índices *k*-grama

- Mais eficiente em relação ao espaço que índice permuterm
- Enumerar todos os *k*-gramas (sequência de *k* caracteres) ocorrendo em um termo
- 2-gramas são chamados bigramas.
- Exemplo: de *April is the cruelest month* temos os bigramas: *$a ap pr ri il l$ $i is s$ $t th he e$ $c cr ru ue el le es st t$ $m mo on nt h$*

Índices *k*-grama

- Mais eficiente em relação ao espaço que índice permuterm
- Enumerar todos os *k*-gramas (sequência de *k* caracteres) ocorrendo em um termo
- 2-gramas são chamados bigramas.
- Exemplo: de *April is the cruelest month* temos os bigramas: *$a ap pr ri il l$ $i is s$ $t th he e$ $c cr ru ue el le es st t$ $m mo on nt h$*
- $ é um símbolo delimitador de palavras, como antes

Índices k-grama

- Mais eficiente em relação ao espaço que índice permuterm
- Enumerar todos os k-gramas (sequência de k caracteres) ocorrendo em um termo
- 2-gramas são chamados bigramas.
- Exemplo: de *April is the cruelest month* temos os bigramas: *$a ap pr ri il l$ $i is s$ $t th he e$ $c cr ru ue el le es st t$ $m mo on nt h$*
- $ é um símbolo delimitador de palavras, como antes
- Manter um índice invertido a partir de bigramas para os termos que têm o bigrama

Lista de índices em um índice invertido com trigramas

índices k-gramas (bigrama, trigrama, . . .)

- Note que agora tem dois diferentes tipos de índices invertidos

índices k-gramas (bigrama, trigrama, . . .)

- Note que agora tem dois diferentes tipos de índices invertidos
- O índice invertido de termos-documentos para encontrar documentos baseados em uma consulta de termos

índices k-gramas (bigrama, trigrama, . . .)

- Note que agora tem dois diferentes tipos de índices invertidos
- O índice invertido de termos-documentos para encontrar documentos baseados em uma consulta de termos
- O índice de k-grams para encontrar termos baseados em uma consulta que consiste de k-grams

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como: $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.
- Nós devemos pós-processar esses termos de acordo com a consulta

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.
- Nós devemos pós-processar esses termos de acordo com a consulta
- Termos restantes são então buscados no índice invertido de termos-documentos

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.
- Nós devemos pós-processar esses termos de acordo com a consulta
- Termos restantes são então buscados no índice invertido de termos-documentos
- índice *k*-gram vs. índice permuterm

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.
- Nós devemos pós-processar esses termos de acordo com a consulta
- Termos restantes são então buscados no índice invertido de termos-documentos
- índice *k*-gram vs. índice permuterm
  - índice *k*-gram é mais eficiente em relação ao espaço

Processar termos com caractere curinga em um índice bigrama

- Consulta mon* agora pode ser executada como:
  $m AND mo AND on
- Obtemos todos termos com o prefixo *mon* . . .
- . . . mas também muitos "falsos positivos" como MOON.
- Nós devemos pós-processar esses termos de acordo com a consulta
- Termos restantes são então buscados no índice invertido de termos-documentos
- índice *k*-gram vs. índice permuterm
  - índice *k*-gram é mais eficiente em relação ao espaço
  - índice permuterm index não requer pós-processamento

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - Muito caro

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - Muito caro
- Problema 2: Usuários odeiam digitar

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - Muito caro
- Problema 2: Usuários odeiam digitar
  - Se consultas como [teor* pitag*] para [teorema de pitágoras] são permitidas, usuários vão usar bastante

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - Muito caro
- Problema 2: Usuários odeiam digitar
  - Se consultas como [teor* pitag*] para [teorema de pitágoras] são permitidas, usuários vão usar bastante
  - Isso potencialmente aumenta o custo de respostas

Processando consultas com curingas no índices de termos-documentos

- Problema 1: precisamos executar um grande número de consultas booleanas
  - Semântica: conjunção de disjunções
  - Para [gen* universit*]: geneva university OR geneva université OR genève university OR genève université OR general universities OR ...
  - Muito caro
- Problema 2: Usuários odeiam digitar
  - Se consultas como [teor* pitag*] para [teorema de pitágoras] são permitidas, usuários vão usar bastante
  - Isso potencialmente aumenta o custo de respostas
- Custo amenizado por recursos do tipo "Google suggest": sugestões durante escrita da busca

Conteúdo

Correção ortográfica

Correção ortográfica

- Dois principais usos

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada
  - Verificar cada palavra individualmente por erros

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada
  - Verificar cada palavra individualmente por erros
  - Não corrige erros que resultam em palavra correta: *um asteroide caiu do seu*

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada
  - Verificar cada palavra individualmente por erros
  - Não corrige erros que resultam em palavra correta: *um asteroide caiu do seu*
- Correção sensível a contexto

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada
  - Verificar cada palavra individualmente por erros
  - Não corrige erros que resultam em palavra correta: *um asteroide caiu do seu*
- Correção sensível a contexto
  - Observa palavras ao redor

Correção ortográfica

- Dois principais usos
  - Correção de documentos na indexação
  - Correção da busca de usuários

Dois métodos para correção

- Correção de palavra isolada
  - Verificar cada palavra individualmente por erros
  - Não corrige erros que resultam em palavra correta: *um asteroide caiu do seu*
- Correção sensível a contexto
  - Observa palavras ao redor
  - pode corrigir *seu*/*céu*

Corrigindo documentos

Corrigindo documentos

- Não estamos interessados em correção iterativa e.g. processador de textos

Corrigindo documentos

- Não estamos interessados em correção iterativa e.g. processador de textos
- Em ORI, usamos correção de documentos principalmente para documentos digitalizados com scanner e OCR

Corrigindo documentos

- Não estamos interessados em correção iterativa e.g. processador de textos
- Em ORI, usamos correção de documentos principalmente para documentos digitalizados com scanner e OCR
- A filosofia básica em ORI é: não alterar documentos

Corrigindo consultas: correção de palavras isoladas

- Premissa 1: existe uma lista de palavras "corretas" da onde vêm as correções

Corrigindo consultas: correção de palavras isoladas

- Premissa 1: existe uma lista de palavras "corretas" da onde vêm as correções
- Premissa 2: temos um meio de computar a distância entre a palavra errada e a correta

Corrigindo consultas: correção de palavras isoladas

- Premissa 1: existe uma lista de palavras "corretas" da onde vêm as correções
- Premissa 2: temos um meio de computar a distância entre a palavra errada e a correta
- Algoritmo de correção simples: retornar a palavra correta que tem a menor distância para a palavra errada

Corrigindo consultas: correção de palavras isoladas

- Premissa 1: existe uma lista de palavras "corretas" da onde vêm as correções
- Premissa 2: temos um meio de computar a distância entre a palavra errada e a correta
- Algoritmo de correção simples: retornar a palavra correta que tem a menor distância para a palavra errada
  - Exemplo: *infomação* $\rightarrow$ *informação*

Corrigindo consultas: correção de palavras isoladas

- Premissa 1: existe uma lista de palavras "corretas" da onde vêm as correções
- Premissa 2: temos um meio de computar a distância entre a palavra errada e a correta
- Algoritmo de correção simples: retornar a palavra correta que tem a menor distância para a palavra errada
  - Exemplo: *infomação* $\rightarrow$ *informação*
- Para a lista de palavras corretas, podemos usar o vocabulário de todas as palavras que ocorrem na nossa coleção

Alternativas para o uso do vocabulário de termos

Alternativas para o uso do vocabulário de termos

- Um dicionário padrão

Alternativas para o uso do vocabulário de termos

- Um dicionário padrão
- Um dicionário especializado

Alternativas para o uso do vocabulário de termos

- Um dicionário padrão
- Um dicionário especializado
  - Exemplo: ORI em advocacia/medicina

Alternativas para o uso do vocabulário de termos

- Um dicionário padrão
- Um dicionário especializado
  - Exemplo: ORI em advocacia/medicina
- Vocabulário de termos da coleção apropriadamente ponderado (pela frequência)

Alternativas para o uso do vocabulário de termos

- Um dicionário padrão
- Um dicionário especializado
  - Exemplo: ORI em advocacia/medicina
- Vocabulário de termos da coleção apropriadamente ponderado (pela frequência)
- Vocabulário de termos obtidos de consultas

Distância entre palavra errada e correta

Distância entre palavra errada e correta

Várias alternativas

Distância entre palavra errada e correta

Várias alternativas

- Distância de edição e de Levenshtein

Distância entre palavra errada e correta

Várias alternativas

- Distância de edição e de Levenshtein
- Distância de edição ponderada

Distância entre palavra errada e correta

Várias alternativas

- Distância de edição e de Levenshtein
- Distância de edição ponderada
- Sobreposição de k-gramas

Distância de edição

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa*-*do*: 1

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa*-*do*: 1
- distância de Levenshtein *caro*-*carro*: 1

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa-do*: 1
- distância de Levenshtein *caro-carro*: 1
- distância de Levenshtein *pato-pata*: 1

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa-do*: 1
- distância de Levenshtein *caro-carro*: 1
- distância de Levenshtein *pato-pata*: 1
- distância de Levenshtein *vôa-avô*: 2

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa-do*: 1
- distância de Levenshtein *caro-carro*: 1
- distância de Levenshtein *pato-pata*: 1
- distância de Levenshtein *vôa-avô*: 2
- distância de Damerau-Levenshtein *cat-act*: 1

Distância de edição

- A distância de edição entre strings s_1 e s_2 é o número mínimo de operações básicas para transformar s_1 em s_2
- Distância de Levenshtein: operações básicas são inserção, remoção e substituição
- distância de Levenshtein *doa-do*: 1
- distância de Levenshtein *caro-carro*: 1
- distância de Levenshtein *pato-pata*: 1
- distância de Levenshtein *vôa-avô*: 2
- distância de Damerau-Levenshtein *cat-act*: 1
- Damerau-Levenshtein inclui transposição como operação básica: *vôa* - *avô*: 1

Distância de Levenshtein

		f	a	s	t
	0	1	2	3	4
c	1	1	2	3	4
a	2	2	1	2	3
t	3	3	2	2	2
s	4	4	3	2	3

Distância de Levenshtein

DISTANCIALEVENSHTEIN(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

Operações: inserir (custo 1), remoção (custo 1), substituir (custo 1), cópia (custo 0)

Distância de Levenshtein

DISTANCIALEVENSHTEIN(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

Operações: inserir (custo 1), remoção (custo 1), substituição (custo 1), cópia (custo 0)

Distância de Levenshtein

DISTANCIALEVENSHTEIN(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

operações: inserção (custo 1), remoção (custo 1), substituição (custo 1), cópia (custo 0)

Distância de Levenshtein

DISTANCIALEVENSHTEIN(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

operações: inserção (custo 1), remoção (custo 1), substituição (custo 1), cópia (custo 0)

Distância de Levenshtein

DISTANCIALEVENSHTEIN(s_1, s_2)

```
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7       do if s1[i] = s2[j]
 8            then m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]}
 9            else  m[i, j] = min{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1}
10  return m[|s1|, |s2|]
```

operações: inserção (custo 1), remoção (custo 1), substituição (custo 1), cópia (custo 0)

Distância de Levenshtein

			f		a		s		t	
		0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
c		**1** **1**	*1* *2*	*2* *1*	**2** **2**	3 **2**	**3** **3**	4 **3**	**4** **4**	5 **4**
a		**2** **2**	**2** 3	**2** **2**	*1* *3*	*3* *1*	*3* *2*	*4* *2*	4 **3**	5 **3**
t		**3** **3**	**3** 4	**3** **3**	3 4	**2** **2**	*2* *3*	*3* *2*	*2* *3*	*4* *2*
s		**4** **4**	**4** 5	**4** **4**	4 5	**3** **3**	**2** 4	3 **2**	*3* *3*	*3* *3*

Cada célula da matriz de Levenshtein

custo de chegar aqui a partir do vizinho superior esquerdo (cópia ou substituição)	custo de chegar aqui a partir do vizinho superior (remoção)
custo de chegar aqui a partir do vizinho da esquerda (inserção)	o mínimo entre os três possíveis "movimentos": o jeito mais barato de chegar aqui

Distância de Levenshtein

			f		a		s		t	
		0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
c		**1** **1**	*1* *2*	*2* *1*	**2** **2**	3 **2**	**3** **3**	4 **3**	**4** **4**	5 **4**
a		**2** **2**	**2** 3	**2** **2**	*1* *3*	*3* *1*	*3* *2*	*4* *2*	4 **3**	5 **3**
t		**3** **3**	**3** 4	**3** **3**	3 4	**2** **2**	*2* *3*	*3* *2*	*2* *3*	*4* *2*
s		**4** **4**	**4** 5	**4** **4**	4 5	**3** **3**	**2** 4	3 **2**	*3* *3*	*3* *3*

Distância de edição ponderada

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.
- Portanto, substituir *m* por *n* é uma distância de edição menor que por *q*.

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.
- Portanto, substituir *m* por *n* é uma distância de edição menor que por *q*.
- Necessita-se de uma matriz de pesos como entrada

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.
- Portanto, substituir *m* por *n* é uma distância de edição menor que por *q*.
- Necessita-se de uma matriz de pesos como entrada
- Como modificar programação dinâmica para manipular pesos?

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.
- Portanto, substituir *m* por *n* é uma distância de edição menor que por *q*.
- Necessita-se de uma matriz de pesos como entrada
- Como modificar programação dinâmica para manipular pesos?
    - Opção 1: se letras são vizinhas no teclado QWERTY, então na substituição distância é 0.5 (arbitrário) senão é 1.

Distância de edição ponderada

- Como antes, mas ponderação depende dos caracteres envolvidos
- Adequado para capturar erros de teclado, e.g., *m* é mais provável de ser escrito errado como *n* que como *q*.
- Portanto, substituir *m* por *n* é uma distância de edição menor que por *q*.
- Necessita-se de uma matriz de pesos como entrada
- Como modificar programação dinâmica para manipular pesos?
    - Opção 1: se letras são vizinhas no teclado QWERTY, então na substituição distância é 0.5 (arbitrário) senão é 1.
    - Opção 2: pesos proporcionais à probabilidade de troca

Usando distância de edição para correção ortográfica

Usando distância de edição para correção ortográfica

- Dada uma consulta, primeiro enumeramos todas as sequências de caracteres dentro de uma distância de edição pré-determinada

Usando distância de edição para correção ortográfica

- Dada uma consulta, primeiro enumeramos todas as sequências de caracteres dentro de uma distância de edição pré-determinada
  - Exemplo: todas as sequências a no máximo 1 unidade de distância de `carra`

Usando distância de edição para correção ortográfica

- Dada uma consulta, primeiro enumeramos todas as sequências de caracteres dentro de uma distância de edição pré-determinada
  - Exemplo: todas as sequências a no máximo 1 unidade de distância de `carra`
- Filtrar as possibilidades pela frequência nos documentos

Usando distância de edição para correção ortográfica

- Dada uma consulta, primeiro enumeramos todas as sequências de caracteres dentro de uma distância de edição pré-determinada
  - Exemplo: todas as sequências a no máximo 1 unidade de distância de `carra`
- Filtrar as possibilidades pela frequência nos documentos
- Então sugerir termos na intersecção ao usuário

Exercício

1. Computar a matriz da distância de Levenshtein para OSLO – SNOW
2. Quais são as operações de edição de Levenshtein que transformam *cat* em *catcat*?

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**				
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	1 2 / 2 ?			
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**			
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	2 3 / 2 ?		
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**		
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	2 4 / 3 ?	
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1**	**1**	2	**2**	3	**2**	4	4	5
	1	2	**1**	**2**	**2**	3	**2**	3	?
s	**2**								
	2								
l	**3**								
	3								
o	**4**								
	4								

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**				
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	1 2 / 3 ?			
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**			
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	2 3 / 2 ?		
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**		
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	3 3 / 3 ?	
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	3 4 / 4 ?
l	**3** / **3**				
o	**4** / **4**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**				
o	**4** / **4**				

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	2 ?						
o	**4** **4**								

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**			
o	**4** / **4**				

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	2 3	3 ?				
o	**4** **4**								

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**				
o	**4** **4**								

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**	3 4 / 3 ?	
o	**4** / **4**				

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**		
o	**4** **4**								

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	4 4	4 ?
o	**4** **4**								

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**								

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	3 ?						

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**						

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	3 4	3 ?				

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**				

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**	**3** 4 / **3** **3**	**4** **4** / **4** **4**
o	**4** / **4**	4 **3** / 5 **3**	**3** **3** / 4 **3**	2 4 / 4 ?	

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**		

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 3	5 ?

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1**	**1**	2	**2**	3	**2**	4	4	5
	1	2	**1**	**2**	**2**	3	**2**	**3**	**3**
s	**2**	**1**	2	**2**	3	**3**	**3**	**3**	4
	2	3	**1**	**2**	**2**	**3**	**3**	4	**3**
l	**3**	3	**2**	**2**	3	**3**	4	**4**	**4**
	3	4	**2**	3	**2**	**3**	**3**	**4**	**4**
o	**4**	4	**3**	**3**	**3**	**2**	4	4	5
	4	5	**3**	4	**3**	4	**2**	**3**	**3**

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** / **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** / **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** / **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**	**3** 4 / **3** **3**	**4** **4** / **4** **4**
o	**4** / **4**	4 **3** / 5 **3**	**3** **3** / 4 **3**	**2** 4 / 4 **2**	4 5 / **3** **3**

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1**	**1**	2	**2**	3	**2**	4	4	5
	1	2	**1**	**2**	**2**	3	**2**	**3**	**3**
s	**2**	**1**	2	**2**	3	**3**	**3**	**3**	4
	2	3	**1**	**2**	**2**	**3**	**3**	4	**3**
l	**3**	3	**2**	**2**	3	**3**	4	**4**	**4**
	3	4	**2**	3	**2**	**3**	**3**	**4**	**4**
o	**4**	4	**3**	**3**	**3**	**2**	4	4	5
	4	5	**3**	4	**3**	4	**2**	**3**	**3**

Como ler as operações de edição que transformam OSLO em SNOW?

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

custo	operação	entrada	saída
1	inserção	*	w

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1** 2	2 **1**	**2** **2**	3 **2**	**2** 3	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2** **2**	3 **2**	**3** **3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3** **3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

custo	operação	entrada	saída
0	(cópia)	o	o
1	inserção	*	w

		s		n		o		w	
		0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1**	**1**	2	**2**	3	**2**	4	4	5
	1	2	**1**	**2**	**2**	3	**2**	**3**	**3**
s	**2**	**1**	2	**2**	3	**3**	**3**	**3**	4
	2	3	**1**	**2**	**2**	**3**	**3**	4	**3**
l	**3**	3	**2**	**2**	3	**3**	4	**4**	**4**
	3	4	**2**	3	**2**	**3**	**3**	**4**	**4**
o	**4**	4	**3**	**3**	**3**	**2**	4	4	5
	4	5	**3**	4	**3**	4	**2**	**3**	**3**

custo	operação	entrada	saída
1	substituição	l	n
0	(cópia)	o	o
1	inserção	*	w

		s	n	o	w
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**
o	**1** **1**	**1** 2 / 2 **1**	**2** 3 / **2** **2**	**2** 4 / 3 **2**	4 5 / **3** **3**
s	**2** **2**	**1** 2 / 3 **1**	**2** 3 / **2** **2**	**3** **3** / **3** **3**	**3** 4 / 4 **3**
l	**3** **3**	3 **2** / 4 **2**	**2** 3 / 3 **2**	**3** 4 / **3** **3**	**4** **4** / **4** **4**
o	**4** **4**	4 **3** / 5 **3**	**3** **3** / 4 **3**	**2** 4 / 4 **2**	4 5 / **3** **3**

custo	operação	entrada	saída
0	(cópia)	s	s
1	substituição	l	n
0	(cópia)	o	o
1	inserção	*	w

		s		n		o		w	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**
o	**1** **1**	**1**	2 **1**	**2**	3 **2**	**2**	4 **2**	4 **3**	5 **3**
s	**2** **2**	**1** 3	2 **1**	**2**	3 **2**	**3**	**3** **3**	**3** 4	4 **3**
l	**3** **3**	3 4	**2** **2**	**2** 3	3 **2**	**3**	4 **3**	**4** **4**	**4** **4**
o	**4** **4**	4 5	**3** **3**	**3** 4	**3** **3**	**2** 4	4 **2**	4 **3**	5 **3**

custo	operação	entrada	saída
1	remoção	o	*
0	(cópia)	s	s
1	substituição	l	n
0	(cópia)	o	o
1	inserção	*	w

		c		a		t		c		a		t	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**
c	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**	6 **5**	7 **5**
a	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**
t	**3** **3**	3 4	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**

		c	a	t	c	a	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**	**5** **5**	**6** **6**
c	**1** / **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**	6 7 / **5** **5**
a	**2** / **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**	5 6 / **4** **4**
t	**3** / **3**	3 **2** / 4 **2**	2 **1** / 3 **1**	**0** 2 / 2 **0**	2 3 / **1** **1**	3 4 / **2** **2**	**3** 5 / **3** **3**

custo	operação	entrada	saída
1	inserção	*	c
1	inserção	*	a
1	inserção	*	t
0	(cópia)	c	c
0	(cópia)	a	a
0	(cópia)	t	t

		c		a		t		c		a		t	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**
c	**1** / **1**	**0** / 2	2 / **0**	2 / **1**	3 / **1**	3 / **2**	4 / **2**	**3** / **3**	5 / **3**	5 / **4**	6 / **4**	6 / **5**	7 / **5**
a	**2** / **2**	2 / 3	**1** / **1**	**0** / **1**	2 / **0**	2 / **1**	3 / **1**	3 / **2**	4 / **2**	**3** / **3**	5 / **3**	5 / **4**	6 / **4**
t	**3** / **3**	3 / 4	**2** / **2**	2 / 3	**1** / **1**	**0** / 2	2 / **0**	2 / **1**	3 / **1**	3 / **2**	4 / **2**	**3** / **3**	5 / **3**

custo	operação	entrada	saída
0	(cópia)	c	c
1	inserção	*	a
1	inserção	*	t
1	inserção	*	c
0	(cópia)	a	a
0	(cópia)	t	t

		c		a		t		c		a		t	
	0	**1**	**1**	**2**	**2**	**3**	**3**	**4**	**4**	**5**	**5**	**6**	**6**
c	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**	6 **5**	7 **5**
a	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**	5 **4**	6 **4**
t	**3** **3**	3 4	**2** **2**	2 3	**1** **1**	**0** 2	2 **0**	2 **1**	3 **1**	3 **2**	4 **2**	**3** **3**	5 **3**

custo	operação	entrada	saída
0	(cópia)	c	c
0	(cópia)	a	a
1	inserção	*	t
1	inserção	*	c
1	inserção	*	a
0	(cópia)	t	t

		c	a	t	c	a	t
	0	**1** **1**	**2** **2**	**3** **3**	**4** **4**	**5** **5**	**6** **6**
c	**1** **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**	5 6 **4** **4**	6 7 **5** **5**
a	**2** **2**	2 **1** 3 **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**	5 6 **4** **4**
t	**3** **3**	3 **2** 4 **2**	2 **1** 3 **1**	**0** 2 2 **0**	2 3 **1** **1**	3 4 **2** **2**	**3** 5 **3** **3**

custo	operação	entrada	saída
0	(cópia)	c	c
0	(cópia)	a	a
0	(cópia)	t	t
1	inserção	*	c
1	inserção	*	a
1	inserção	*	t

Conteúdo

Correção ortográfica

Correção ortográfica

- Agora que sabemos computar a distância de edição: como usá-la para correção ortográfica de palavras isoladas

Correção ortográfica

- Agora que sabemos computar a distância de edição: como usá-la para correção ortográfica de palavras isoladas
- Índices k-grams para correção ortográfica de palavras isoladas

Correção ortográfica

- Agora que sabemos computar a distância de edição: como usá-la para correção ortográfica de palavras isoladas
- Índices k-grams para correção ortográfica de palavras isoladas
- Correção ortográfica sensível a contexto

Índices de k-gramas indexes para correção ortográfica

Índices de k-gramas indexes para correção ortográfica

- Enumerar todos k-gramas no termo de consulta

Índices de k-gramas indexes para correção ortográfica

- Enumerar todos k-gramas no termo de consulta
  - Exemplo: índice com bigramas, palavra errada *bordroom*

Índices de k-gramas indexes para correção ortográfica

- Enumerar todos k-gramas no termo de consulta
  - Exemplo: índice com bigramas, palavra errada *bordroom*
  - Bigramas: *bo, or, rd, dr, ro, oo, om*

Índices de *k*-gramas indexes para correção ortográfica

- Enumerar todos *k*-gramas no termo de consulta
  - Exemplo: índice com bigramas, palavra errada *bordroom*
  - Bigramas: *bo, or, rd, dr, ro, oo, om*
- Usar o índice *k*-grama para recuperar palavras "corretas" que correspondem ao termo *k*-grams

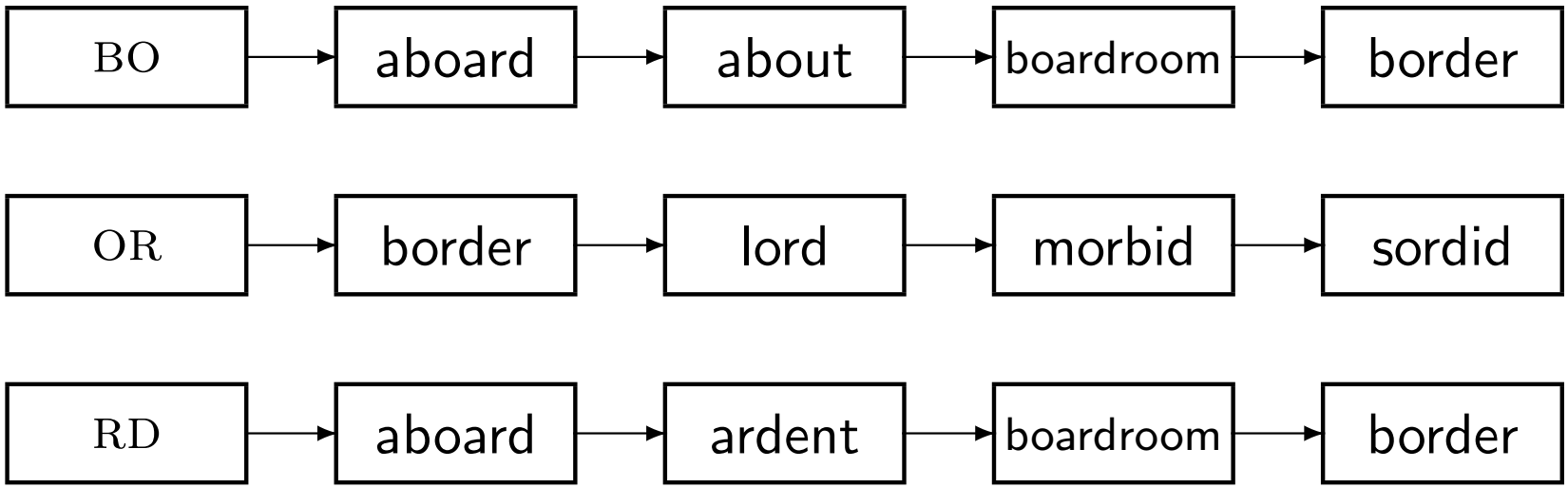
Índices de *k*-gramas indexes para correção ortográfica

- Enumerar todos *k*-gramas no termo de consulta
  - Exemplo: índice com bigramas, palavra errada *bordroom*
  - Bigramas: *bo, or, rd, dr, ro, oo, om*
- Usar o índice *k*-grama para recuperar palavras "corretas" que correspondem ao termo *k*-grams
- Limitar o número de *k*-gramas para considerar como candidato

Índices de *k*-gramas indexes para correção ortográfica

- Enumerar todos *k*-gramas no termo de consulta
  - Exemplo: índice com bigramas, palavra errada *bordroom*
  - Bigramas: *bo, or, rd, dr, ro, oo, om*
- Usar o índice *k*-grama para recuperar palavras "corretas" que correspondem ao termo *k*-grams
- Limitar o número de *k*-gramas para considerar como candidato
  - Exemplo: somente termos do vocabulário que diferem em até 3 *k*-gramas

Índices *k*-gramas para correção ortográfica: *bordroom*



Correção ortográfica sensível a contexto

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*
  - Fazer internamente todas as consultas resultantes com uma palavra "corrigida" por vez

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*
  - Fazer internamente todas as consultas resultantes com uma palavra "corrigida" por vez
    - Consulte *"asteroide" "caiu" "terra"*

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*
  - Fazer internamente todas as consultas resultantes com uma palavra "corrigida" por vez
    - Consulte *"asteroide" "caiu" "terra"*
    - Consulte *"esteroide" "caio" "terra"*

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*
  - Fazer internamente todas as consultas resultantes com uma palavra "corrigida" por vez
    - Consulte *"asteroide" "caiu" "terra"*
    - Consulte *"esteroide" "caio" "terra"*
    - Consulte *"esteroide" "caiu" "terá"*

Correção ortográfica sensível a contexto

- Exemplo: *esteroide caiu terra*
- Podemos corrigir esteroide?
- Ideia: correção ortográfica baseada em número de resultados
  - Recuperar termos "corretos" mais próximos a cada termo de consulta
    - *esteroide* $\rightarrow$ *asteroide*, *caiu* $\rightarrow$ *caio*, *terra* $\rightarrow$ *terá*
  - Fazer internamente todas as consultas resultantes com uma palavra "corrigida" por vez
    - Consulte *"asteroide" "caiu" "terra"*
    - Consulte *"esteroide" "caio" "terra"*
    - Consulte *"esteroide" "caiu" "terá"*
  - A consulta correta *"asteroide caiu terra"* (com sorte) terá o maior número de resultados

Correção ortográfica sensível a contexto

Correção ortográfica sensível a contexto

- O algoritmo baseado em número de resultados não é eficiente

Correção ortográfica sensível a contexto

- O algoritmo baseado em número de resultados não é eficiente
- Alternativa mais eficiente: usar coleção de consultas e não documentos

Problemas gerais na correção ortográfica

Problemas gerais na correção ortográfica

- Interface de usuário

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista
- Custo

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista
- Custo
  - Correção ortográfica é potencialmente cara

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista
- Custo
  - Correção ortográfica é potencialmente cara
  - Evitar executar em toda consulta

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista
- Custo
  - Correção ortográfica é potencialmente cara
  - Evitar executar em toda consulta
  - Talvez somente em consultas com poucos documentos

Problemas gerais na correção ortográfica

- Interface de usuário
  - Correção automática vs. correção sugerida
  - *Você quis dizer:* somente uma sugestão
  - E múltiplas correções possíveis?
  - Balanço: interface simples ou detalhista
- Custo
  - Correção ortográfica é potencialmente cara
  - Evitar executar em toda consulta
  - Talvez somente em consultas com poucos documentos
  - Correção ortográfica dos maiores buscadores é eficiente para ser realizada em toda consulta

Conteúdo

Soundex

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)
- Exemplo: *chebyshev* / *tchebyscheff*

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)
- Exemplo: *chebyshev* / *tchebyscheff*
- Algoritmo:

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)
- Exemplo: *chebyshev* / *tchebyscheff*
- Algoritmo:
  - Transformar cada token indexado em uma forma reduzida de 4-caracteres

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)
- Exemplo: *chebyshev* / *tchebyscheff*
- Algoritmo:
  - Transformar cada token indexado em uma forma reduzida de 4-caracteres
  - Faça o mesmo com termos de consulta

Soundex

- Soundex é a base para encontrar alternativas fonéticas (e não ortográficas)
- Exemplo: *chebyshev* / *tchebyscheff*
- Algoritmo:
  - Transformar cada token indexado em uma forma reduzida de 4-caracteres
  - Faça o mesmo com termos de consulta
  - Construir e procurar em um índice de formas reduzidas

Algoritmo Soundex (inglês)

1. Reter a primeira letra do termo
2. Mudar todas as ocorrências das letras seguintes para '0' (zero): A, E, I, O, U, H, W, Y
3. Mudar letras para dígitos da seguinte forma:
   - B, F, P, V para 1
   - C, G, J, K, Q, S, X, Z para 2
   - D,T para 3
   - L para 4
   - M, N para 5
   - R para 6
4. Remover repetidamente um de cada par de dígitos idênticos
5. Remover todos os zeros da string resultante
6. Completar a string resultante com zeros
7. Retornar as quatro primeiras posições, que consistirão de uma letra seguida por três dígitos

Exemplo: Soundex de *HERMAN*

Exemplo: Soundex de *HERMAN*

- Reter H

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*
- *0RM0N* $\rightarrow$ *06505*

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*
- *0RM0N* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *06505*

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*
- *0RM0N* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *655*

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*
- *0RM0N* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *655*
- Retornar *H655*

Exemplo: Soundex de *HERMAN*

- Reter H
- *ERMAN* $\rightarrow$ *0RM0N*
- *0RM0N* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *06505*
- *06505* $\rightarrow$ *655*
- Retornar *H655*
- Note: *HERMANN* gerará o mesmo código

O quão útil é Soundex?

O quão útil é Soundex?

- Útil para tratar nomes

O quão útil é Soundex?

- Útil para tratar nomes
- Ok para tarefas de "alta recuperação" em aplicações específicas (e.g., Interpol)