

Carlos Henrique Pacheco de Souza

Análise Exploratória de Dados em Redes de Compartilhamento de Conhecimento

Belo Horizonte

2017/1

Carlos Henrique Pacheco de Souza

Análise Exploratória de Dados em Redes de Compartilhamento de Conhecimento

Apresentado como requisito da disciplina de
Monografia em Sistemas de Informação do
DCC / UFMG.

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Orientador: Clodoveu Augusto Davis Junior
Departamento de Ciência da Computação

Belo Horizonte

2017/1

Resumo

O desenvolvimento de software apresenta grandes desafios que vão além do conhecimento técnico. Fatores importantes como os aspectos sociais e culturais são ignorados. Compreender como está a distribuição do conhecimento técnico nas diferentes regiões tem grande importância para definir o sucesso do projeto.

Com o surgimento de redes sociais como StackOverflow e GitHub, uma vasta quantidade de informações é criada diariamente. A grande utilização destas redes no desenvolvimento de software, permite que seja realizada uma nova abordagem na análise de seus dados.

Para isso, utilizaremos dados geográficos que serão obtidos através da geocodificação. Endereços urbanos são uma das principais formas de expressão da localização geográfica em cidades. Muitos sistemas de informação incluem atributos para receber endereços e, assim, contam com uma referência espacial indireta. A obtenção de coordenadas a partir de endereços é um dos métodos de geocodificação mais importantes, mas é dificultada por variações comuns no endereço, como abreviações e omissão de componentes.

Temos como objetivo principal, criar uma estrutura que amplie o poder de informação das bases de dados selecionadas. E assim, permitir que dados geográficos sejam utilizados nas pesquisas, tanto nos filtros quanto na visualização das respostas.

Palavras-chaves: desenvolvimento de software, StackOverflow, GitHub, geocodificação.

Lista de ilustrações

Figura 1 – Projeto disponibilizao no GitHub.	9
Figura 2 – Script: 010-download.sh	9
Figura 3 – Script: 020-extracao.sh	10
Figura 4 – Script: Algoritmo R0	11
Figura 5 – Script: Algoritmo R1	11
Figura 6 – Script: Algoritmo R2	12
Figura 7 – Script: Algoritmo R3	13
Figura 8 – Diagrama de Entidade e Relacionamento	13
Figura 9 – Script: 050-modelagem.sh	14
Figura 10 – Interface para Visualização das Informações	14
Figura 11 – Usuários das Redes	15
Figura 12 – Usuários Git Hub	15
Figura 13 – Usuários Stack Overflow	16
Figura 14 – Resultado: Algoritmo R0	16
Figura 15 – Resultado: Algoritmo R1	17
Figura 16 – Resultado: Algoritmo R2	17
Figura 17 – Resultado: Algoritmo R3	18
Figura 18 – Nuvem de Palavras sem Localização	18

Sumário

1	INTRODUÇÃO	5
2	REFERENCIAL TEÓRICO	7
3	METODOLOGIA	9
3.1	Obtenção dos Dados	9
3.2	Extração das Informações	10
3.3	Processamento	10
3.3.1	Algoritmo R0 - Localização por Sigla	10
3.3.2	Algoritmo R1 - Localização por Correspondência Completa de Nomes . . .	11
3.3.3	Algoritmo R2 - Localização por Correspondência Completa de Topônimos .	12
3.3.4	Algoritmo R3 - Localização por Distância Editável	12
3.4	Modelagem	13
3.5	Visualização	14
4	ANÁLISES E RESULTADOS	15
5	CONCLUSÃO	19
	REFERÊNCIAS	20

1 Introdução

O desenvolvimento de software apresenta grandes desafios que vão além do conhecimento técnico. Um problema muito comum ocorre durante a escolha de quais tecnologias e ferramentas utilizar. Diversos estudos são realizados na busca da combinação perfeita. Porém, alguns fatores importantes como os aspectos sociais e culturais são ignorados.

Nenhum sistema é criado sem a utilização de mão de obra especializada. Assim, o custo do projeto e consequentemente o seu sucesso, são influenciados pela demanda desses profissionais. Portanto, compreender como está a distribuição do conhecimento técnico nas diferentes regiões tem grande importância para essa tomada de decisão.

A grande utilização das redes sociais de compartilhamento de conhecimento no desenvolvimento de software, permite que seja realizada uma nova abordagem na análise de seus dados. Para isso, utilizaremos dados geográficos na visualização do comportamento de duas das maiores redes que foram selecionadas para esse trabalho devido não apenas aos seus tamanhos mas também a sua complementaridade proposta por Vasilescu, Filkov e Serebrenik (2013).

- GitHub é um serviço de hospedagem web para projetos que usam o controle de versionamento Git. Ao contrário do Git, que é uma ferramenta que opera estritamente na linha de comando, o GitHub fornece uma interface gráfica baseada na Web, bem como a integração com dispositivos móveis. Ele também fornece controle de acesso a vários recursos de colaboração, tais como rastreamento de bugs, solicitações de recursos, gerenciamento de tarefas e wikis por projeto. Em 2016 ele conta com aproximadamente 14 milhões de usuários e mais de 35 milhões de repositórios, tornando-se o maior servidor de código fonte no mundo(WIKIPEDIA, 2016a).
- Stack Overflow é um site que apresenta perguntas e respostas numa grande quantidade de tópicos de programação de computadores. Ele serve como plataforma para que os usuários façam perguntas e também as respondam. Mas além disso, podem através do registro e da participação ativa, que os usuários votem em questões e respostas mais ou menos úteis. O fechamento de perguntas é a principal diferenciação da ferramenta que previne perguntas com baixa qualidade. Em 2016 ele conta com aproximadamente 6 milhões de usuários e mais de 11 milhões de perguntas(WIKIPEDIA, 2016b).

Temos como objetivo principal, criar uma estrutura que amplie o poder de informação das bases de dados selecionadas. E assim, permitir que dados geográficos sejam utilizados nas pesquisas, tanto nos filtros quanto na visualização das respostas.

Como objetivo secundário realizaremos uma breve análise sobre os dados, identificando os lugares onde cada uma das redes sociais possui maior presença relativa. Bem como, o

comportamento dos usuários que informaram sua localização. Dentre outras análises que também são possíveis, podemos destacar a concentração relativa dos usuários por tópicos e ou períodos de interação nas redes. Além de identificar as regiões que produziram informações (códigos e ou respostas) de melhor qualidade. Para mensurar este fator, podemos utilizar o índice CPDScorer (HUANG et al.,).

2 Referencial Teórico

Com o surgimento de redes sociais como StackOverflow e GitHub, uma vasta quantidade de informações de desenvolvimento é criada diariamente. Tais dados de contexto pessoal e social tem um enorme potencial para apoiar na avaliação automática e eficaz da capacidade do desenvolvedor. O CPDScorer aborda uma modelagem e avaliação da capacidade de programação dos desenvolvedores através da mineração de informações heterogêneas, presentes nas comunidades do StackOverflow e GitHub. CPDScorer considera tanto a qualidade de resposta sobre tópicos de programação do StackOverflow e na qualidade do código fonte do projeto no GitHub. Um algoritmo de extração de termos da capacidade de programação também é projetado para rotular cada resposta e projeto com um termo de habilidade (HUANG et al.,).

A complementaridade das ferramentas foi constatada num estudo que mostra que "committers" ativos do GitHub perguntam menos e fornecem mais respostas no StackOverflow. Assim como, "askers" ativos do StackOverflow distribui seu trabalho de uma maneira menos uniforme que os outros (VASILESCU; FILKOV; SEREBRENIK, 2013). Bem como comportamentos semelhantes entre usuários das redes diferentes foram relatados por Badashian et al. (2014) ao verificar que desenvolvedores ativos contribuem para as principais atividades da plataforma (ou seja, realizar commits no GitHub e responder no StackOverflow), mas também se envolvem em outras atividades gerenciais (como gerenciamento de problemas no GitHub e voto de qualidade no StackOverflow).

Geocodificação é um conjunto de métodos capazes de transformar descrições em coordenadas geográficas. Endereços urbanos são uma das principais formas de expressão da localização geográfica em cidades. Muitos sistemas de informação incluem atributos para receber endereços e, assim, contam com uma referência espacial indireta. A obtenção de coordenadas a partir de endereços é um dos métodos de geocodificação mais importantes, mas é dificultada por variações comuns no endereço, como abreviações e omissão de componentes. No caso de endereços, existe uma expectativa de detalhamento hierárquico, com componentes que indicam o país, o estado e a cidade. O formato de apresentação desses componentes varia de país para país, e em muitas situações, alguns componentes são intencionalmente omitidos ou simplificados(MARTINS; JR; FONSECA, 2012).

Para contornar essa variabilidade na formação dos endereços, uma solução consiste na divisão do método em três passos ou estágios, sendo que cada estágio possui tarefas e interfaces de entrada e saída bem definidas. O primeiro estágio, chamado de "parsing", consiste na análise léxica que leve em conta as peculiaridades da estrutura de endereços do local ou país e posterior conversão da entrada textual contendo o endereço em uma estrutura de dados genérica. Essa estrutura de dados contém um número finito de atributos, que correspondem a

cada componente do endereço. O segundo estágio, chamado de "matching", recebe a estrutura de dados e realiza buscas em um banco de dados de referência, comparando valores por casamento exato ou aproximado de palavras e números, e definindo a melhor solução em caso de casamento parcial. Nesta fase, utiliza-se Levenshtein distance (NAVARRO, 2001) e Shift-And(WU; MANBER, 1992). O estágio seguinte, denominado "locating", consiste em recuperar as referências obtidas e extrair delas as coordenadas desejadas(JR; FONSECA, 2007).

3 Metodologia

O projeto foi dividido em 6 fases que são sequenciais e, seguem a ordem em que foram postadas a seguir. Ele está disponível em <https://github.com/krloss/ufmg-msi-rcc>:

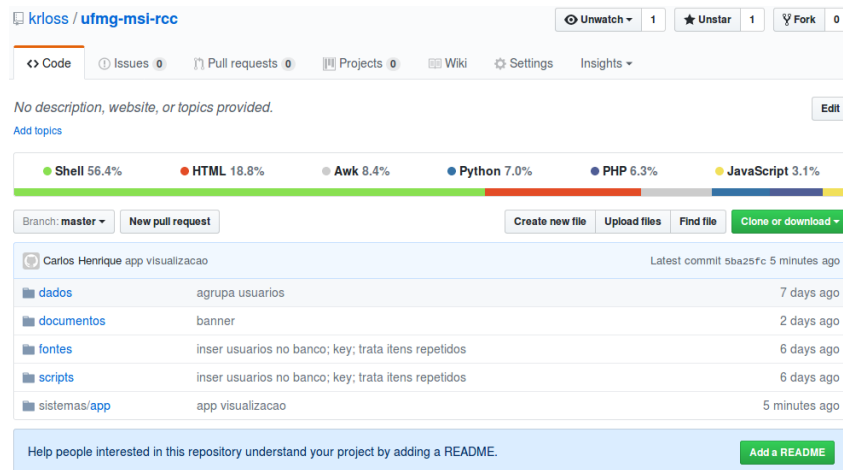


Figura 1 – Projeto disponibilizado no GitHub.

3.1 Obtenção dos Dados

Inicialmente pensamos em utilizar os dados diretamente das interfaces de programação de aplicativos (APIs - StackExchange¹ e Geonames²) e dos bancos de dados disponíveis na internet (GH Torrent³). Porém, a grande demanda de informações durante o processamento, exigiram que os dados fossem acessados localmente. Entretanto, como os datasets são extensos, tivemos que filtrar as informações que foram baixadas através do seguinte script:

```
Terminal
~/bin/sh
cd ../fontes/

wget -b https://archive.org/download/stackexchange/stackoverflow.com-Users.7z

ssh-keygen -t rsa -b 2048 -C "carloss@ufmg.br" # ./id_rsa
# GitHub fork project https://github.com/ghorrent/ghorrent.org.git
git clone https://github.com/krloss/ghorrent.org.git; cd ghorrent.org/
cat ../id_rsa.pub >> keys.txt; git commit -am 'Add key'; git push
# Create pull request
cd ../

ssh -i ssh_rsa_key -NfL *:8017:dutibr.st.ewi.tudelft.nl:27017 ghorrent@dutibr.st.ewi.tudelft.nl
mongoexport -u ghorrentro -p ghorrentro -d github -c users -q '{location:/\\S(2,)/}' -f id,location -o github-users.json

wget -b http://download.geonames.org/export/dump/allCountries.zip
wget -b http://download.geonames.org/export/dump/countryInfo.txt
wget -b http://download.geonames.org/export/dump/hierarchy.zip
```

Figura 2 – Script: 010-download.sh

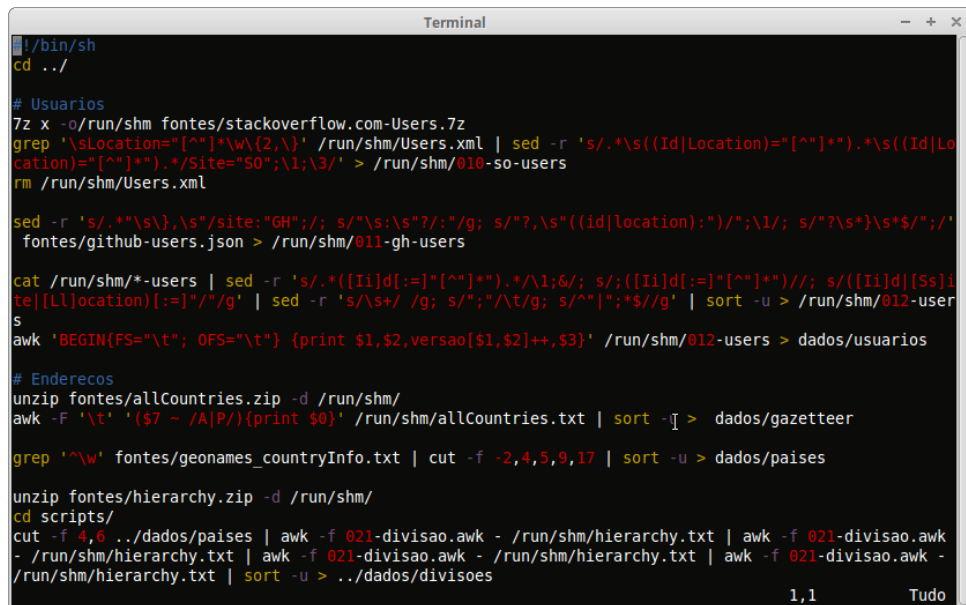
¹ <https://data.stackexchange.com/stackoverflow/query/new>

² <http://api.geonames.org/>

³ <http://ghorrent.org/dblite>

3.2 Extração das Informações

Esta fase é responsável por mapear e extrair das base de dados as informações que serão utilizadas. Aqui, filtramos todos os usuários que informaram pelo menos dois caracteres no campo de localização (aproximadamente 10% de todos os usuários cadastrados). Buscamos também, os endereços de lugares como cidade, estado e país.



```
#!/bin/sh
cd ../

# Usuarios
7z x -o/run/shm fontes/stackoverflow.com-Users.7z
grep '\sLocation="[^\"]*\w\{2,\}' /run/shm/Users.xml | sed -r 's/.*\s((Id|Location)="[^\"]*").*\s((Id|Location)="[^\"]*").*/Site="SO";\1;\3/' > /run/shm/010-so-users
rm /run/shm/Users.xml

sed -r 's/.*"\s\{,}\s"/site:"GH";/; s/"\s\s?"/:/g; s/"?\s"({id|location}:)"/"/\1/; s/"?\s\{,}\s$"/;/; fontes/github-users.json > /run/shm/011-gh-users

cat /run/shm/*-users | sed -r 's/.*([Ii]d[:=]"[^\"]*").*/\1&/; s/;([Ii]d[:=]"[^\"]*")//; s/([Ii]d)[Ss]l te|[Ll]ocation)[:=]"[^\"]*"/g | sed -r 's/\s+/ /g; s/"/"/\t/g; s/^\s*"|";*$/g' | sort -u > /run/shm/012-user
s
awk 'BEGIN{FS="\t"; OFS="\t"} {print $1,$2,versao[$1,$2]++,$3}' /run/shm/012-users > dados/usuarios

# Enderecos
unzip fontes/allCountries.zip -d /run/shm/
awk -F '\t' '($7 ~ /A|P/){print $0}' /run/shm/allCountries.txt | sort -t > dados/gazetteer

grep '^w' fontes/geonames_countryInfo.txt | cut -f -2,4,5,9,17 | sort -u > dados/paises

unzip fontes/hierarchy.zip -d /run/shm/
cd scripts/
cut -f 4,6 ../dados/paises | awk -f 021-divisao.awk - /run/shm/hierarchy.txt | awk -f 021-divisao.awk - /run/shm/hierarchy.txt | awk -f 021-divisao.awk - /run/shm/hierarchy.txt | sort -u > ../dados/divisooes
```

Figura 3 – Script: 020-extracao.sh

3.3 Processamento

Neste momento iremos processar os dados em busca da localização geográfica dos usuários. Fase em que utilizaremos conceitos identificados no referencial teórico. Ela está dividida em 4 partes que seguem abaixo:

3.3.1 Algoritmo R0 - Localização por Sigla

O primeiro algoritmo executado, visa identificar o endereço do usuário que digitou até 3 caracteres no campo localização. Por se tratar de sites internacionais, entendemos que estes endereços possam ser a sigla de algum país de acordo com um estudo empírico sobre os dados.

```

Terminal
#!/bin/sh
cd ../dados/
ln -rs ../run/shm/origem
awk 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $4); if(length($4) > 1) print toupper($0)}' usuarios > localizacao
cut -f 4 localizacao | grep '[A-Z]' | sort -u > /run/shm/020-locais

# Algoritmo R0: Encontra localização por sigla do país(casamento exato)
awk 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR){m3[$2]=$6; m2[2 == length($3) ? $3 : $1]=$6} (NR!=FNR && $0 in m3){print $0, m3[$0]} (NR!=FNR && $0 in m2){print $0, m2[$0]} pais /run/shm/020-locais > R0
awk -F '\t' ' (NR==FNR){mapa[$1]} (NR!=FNR && !($1 in mapa)){print $0}' R0 /run/shm/020-locais > /run/shm/021-locais

```

Figura 4 – Script: Algoritmo R0

3.3.2 Algoritmo R1 - Localização por Correspondência Completa de Nomes

Este algoritmo, visa identificar o endereço do usuário através da igualdade de nomes. Desse modo, o algoritmo seleciona **usuários** que tenham aquela unidade de nome e aplica uma nota(peso) para verificar o quanto que a unidade representa de todo o texto. No final, o algoritmo favorece nomes que estejam na mesma hierarquia. Por exemplo, quando o usuário informa a localização "*Betim - Minas Gerais*". No mínimo existem 3 localizações possíveis:

1. Betim(Asia / Kolkata) com nota: $5/20 = 0,25$
2. Betim(América / São Paulo) com nota: $5/20 = 0,25$
3. Minas Gerais(América / São Paulo) com nota: $12/20 = 0,60$

Analisando a hierarquia descobrimos que o segundo item "*Betim(América / São Paulo)*" pertence ao terceiro "*Minas Gerais(América / São Paulo)*". Então a nota dele é **somada** a nota do item pai, resultando em 0,85.

```

Terminal
awk 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes_gazetteer > /run/shm/030-gazetteer
awk -v adm="(.)" {4} {1} 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR && $0 ~ adm){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes
/run/shm/030-gazetteer | sort -u > /run/shm/040-divisao
awk -v adm="(.)" {3} {1} 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR && $0 ~ adm){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes
/run/shm/030-gazetteer | sort -u > /run/shm/041-divisao
awk -v adm="(.)" {2} {1} 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR && $0 ~ adm){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes
/run/shm/030-gazetteer | sort -u > /run/shm/042-divisao
awk -v adm="(.)" {1} {1} 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR && $0 ~ adm){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes
/run/shm/030-gazetteer | sort -u > /run/shm/043-divisao
awk -v adm="(.)" {0} {1} 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR && $0 ~ adm){mapa[$2]} (NR!=FNR && $1 in mapa){print $0}' divisoes
/run/shm/030-gazetteer | sort -u > /run/shm/044-divisao

# Algoritmo R1: Encontra localização por regex em nomes(correspondência completa)
awk -v nome=1 -v id=0 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $nome); local=toupper($nome); system("grep \"^\"local\" /run/shm/021-locais | sed \"s/$/\\t\"$id\"length(local)\"/\"")}' pais > /run/shm/050-N0
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $nome); local=toupper($nome); system("grep \"^\"local\" /run/shm/021-locais | sed \"s/$/\\t\"$id\"length(local)\"/\"")}' /run/shm/041-divisao > /run/shm/050-N1
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $nome); local=toupper($nome); system("grep \"^\"local\" /run/shm/021-locais | sed \"s/$/\\t\"$id\"length(local)\"/\"")}' /run/shm/042-divisao > /run/shm/050-N2
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $nome); local=toupper($nome); system("grep \"^\"local\" /run/shm/021-locais | sed \"s/$/\\t\"$id\"length(local)\"/\"")}' /run/shm/043-divisao > /run/shm/050-N3
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS="\t"} {gsub(/W/, "", $nome); local=toupper($nome); system("grep \"^\"local\" /run/shm/021-locais | sed \"s/$/\\t\"$id\"length(local)\"/\"")}' /run/shm/044-divisao > /run/shm/050-N4

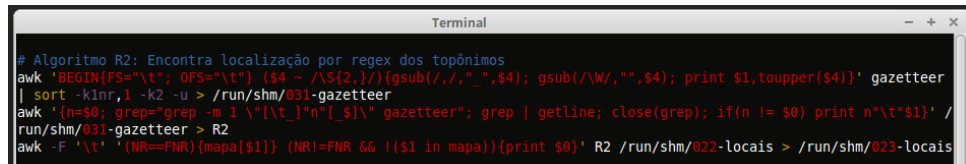
cd /run/shm/
awk -f origem/scripts/031-criterio.awk origem/dados/divisoes 050-N* > 051-criterios
cd origem/dados/
grep -VP '\t0\.[01][20-4][0-9]*\t' /run/shm/051-criterios | sort -k1,1 -k2r,2 -k3,3 -k4r | sort -uk 1,1 > R1
awk -F '\t' ' (NR==FNR){mapa[$1]} (NR!=FNR && !($1 in mapa)){print $0}' R1 /run/shm/021-locais > /run/shm/022-locais

```

Figura 5 – Script: Algoritmo R1

3.3.3 Algoritmo R2 - Localização por Correspondência Completa de Topônimos

Este algoritmo, visa identificar o endereço do usuário através da igualdade de nomes. Desse modo, o algoritmo seleciona **endereços** que tenham aquela unidade de nome no campo dos topônimos. Aqui, encontramos nomes que são escritos em outras línguas ou possuem alguma variação.



```
Terminal
# Algoritmo R2: Encontra localização por regex dos topônimos
awk 'BEGIN(FS="\t"; OFS="\t") { $4 ~ /\S{2,}/ } { gsub(/./, "_", $4); gsub(/W/, "", $4); print $1, toupper($4) } ' gazetteer
| sort -k1nr,1 -k2 -u > /run/shm/031-gazetteer
awk '{ n=$0; grep="grep -m 1 \"[t_]*n\"[t_]*" gazetteer"; grep | getline; close(grep); if(n != $0) print n"\t"$1' /
run/shm/031-gazetteer > R2
awk -F '\t' ' (NR==FNR){mapa[$1]} (NR!=FNR && !($1 in mapa)){print $0}' R2 /run/shm/022-locais > /run/shm/023-locais
```

Figura 6 – Script: Algoritmo R2

3.3.4 Algoritmo R3 - Localização por Distância Editável

O último algoritmo executado, visa identificar o endereço do usuário utilizando Levenshtein distance (NAVARRO, 2001). Desse modo, o algoritmo seleciona **usuários** que tenham informado uma localização semelhante aquela unidade de nome e aplica uma nota(distância) para verificar o quanto que a unidade está distante de todo o texto. No final, o algoritmo favorece nomes que estejam na mesma hierarquia. Por exemplo, quando o usuário informa a localização "Btim - Mina Graís". No mínimo existem 3 localizações possíveis:

1. Betim(Ásia / Kolkata) com nota: $14/17 = 0,82$
2. Betim(África / São Paulo) com nota: $14/17 = 0,82$
3. Minas Gerais(África / São Paulo) com nota: $9/17 = 0,53$

Analisando a hierarquia descobrimos que o segundo item "Betim(África / São Paulo)" pertence ao terceiro "Minas Gerais(África / São Paulo)". Então a nota dele é **multiplicada** a nota do item pai, resultando em 0,44.

```

Terminal
awk -v nome=4 -v id=6 'BEGIN{FS="\t"; OFS=" "} {gsub(/\W/, "", $nome); print $id,toupper($nome)}' paises > /run/shm/845-D0
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS=" "} {gsub(/\W/, "", $nome); print $id,toupper($nome)}' /run/shm/841-divisao > /run/shm/845-D1
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS=" "} {gsub(/\W/, "", $nome); print $id,toupper($nome)}' /run/shm/842-divisao > /run/shm/845-D2
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS=" "} {gsub(/\W/, "", $nome); print $id,toupper($nome)}' /run/shm/843-divisao > /run/shm/845-D3
awk -v nome=2 -v id=1 'BEGIN{FS="\t"; OFS=" "} {gsub(/\W/, "", $nome); print $id,toupper($nome)}' /run/shm/844-divisao > /run/shm/845-D4

# Algoritmo R3: Encontra localização através da distância editável
cd ../scripts/
pip install -t distEditavel/ nltk
distEditavel/ned.py -- /run/shm/823-locais /run/shm/845-D0 > /run/shm/868-N0
distEditavel/ned.py -- /run/shm/823-locais /run/shm/845-D1 > /run/shm/868-N1
split /run/shm/845-D2 -l 21278 /run/shm/846-D2
split /run/shm/845-D3 -l 40000 /run/shm/846-D3
split /run/shm/845-D4 -l 38226 /run/shm/846-D4
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D2aa > /run/shm/861-N2a
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D2ab > /run/shm/861-N2b
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D3aa > /run/shm/861-N3a
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D3ab > /run/shm/861-N3b
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D4aa > /run/shm/861-N4a
distEditavel/ned.py -- /run/shm/823-locais /run/shm/846-D4ab > /run/shm/861-N4b
cat /run/shm/861-N2* | sort -k1,1 -k3n,3 -k2nr,2 | sort -uk1,1 > /run/shm/868-N2
cat /run/shm/861-N3* | sort -k1,1 -k3n,3 -k2nr,2 | sort -uk1,1 > /run/shm/868-N3
cat /run/shm/861-N4* | sort -k1,1 -k3n,3 -k2nr,2 | sort -uk1,1 > /run/shm/868-N4
cd /run/shm/
awk -f origem/scripts/832-distancia.awk origem/dados/divisoes 868-N* > 852-distancias
cd origem/dados/
grep -P '\to\.[0-9]{0-4}[0-9]{0-9}' /run/shm/852-distancias | sort -k1,2 -k4r | sort -uk1,1 > R3
awk -F '\t' '{(NR==FNR){mapa[$1]} (NR!=FNR){mapa[$1]} (print $0)}' R3 /run/shm/823-locais > _R
38,0-1 Fim

```

Figura 7 – Script: Algoritmo R3

3.4 Modelagem

No início armazenamos o resultado do processamento num banco de dados não relacional(MongoDB). Mas devido as limitações para acessa-lo, alteramos o resultado para o MySQL.

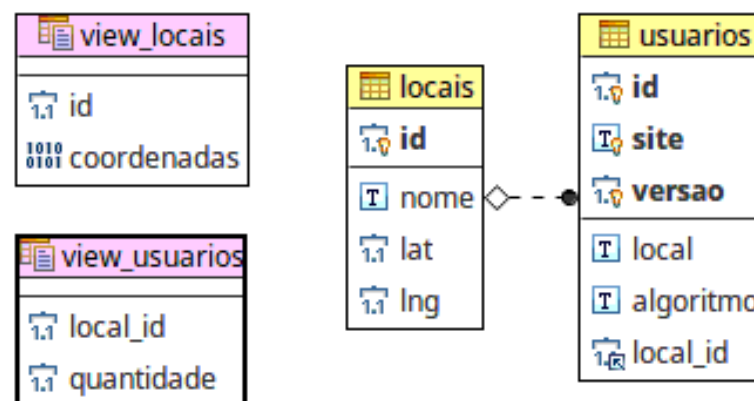


Figura 8 – Diagrama de Entidade e Relacionamento

A estrutura acima permite a realização de consultas utilizando recursos de banco de dados geográfico, como por exemplo, buscar localizações dentro de uma área geométrica.

```

SELECT * FROM locais l, view_locais vl WHERE l.id=vl.id AND Contains(GeomFromText(
'POLYGON((-26.5 -46.5,-17 -36,-12.5 -47.0,-22.5 -53.5,-26.5 -46.5))'),coordenadas);

```

Para preencher a estrutura criada, utilizamos o seguinte script:

```

Terminal
#!/bin/sh
cd ../dados/

echo 'INSERT INTO locais (id,nome,lat,lng) VALUES' > locais.sql
cat R? | grep -o '\S+\s*$' | sort -u | awk -v aa="" 'BEGIN{FS="\t"; OFS=","} (NR==FNR){mapa[$1]
} (NR!=FNR && $1 in mapa){print "(" $1,aa $2 aa,$5,$6"),"}' - gazetteer >> locais.sql

echo 'INSERT INTO usuarios (id,site,versao,local,algoritmo,local_id) VALUES' > usuarios.sql
awk 'BEGIN{FS="\t"; OFS="\t"} {print $1,FILENAME,$NF}' R? | sort > /run/shm/070-mapa
awk 'BEGIN{FS="\t"; OFS="\t"} (NR==FNR){mapa[$1]=$0} (NR!=FNR && $4 in mapa){print $1,$2,$3,mapa[
$4]}' /run/shm/070-mapa localizacao > /run/shm/071-mapa
awk -v aa="" 'BEGIN{FS="\t"; OFS=","} (NR==FNR){mapa[$1,$2,$3]=aa $5 aa"," $6} (NR!=FNR){m=mapa[
$1,$2,$3]; print "(" $1,aa $2 aa,$3,aa $4 aa,(m ? m : "NULL,NULL")"),"}' /run/shm/071-mapa usuarios
>> usuarios.sql

```

Figura 9 – Script: 050-modelagem.sh

3.5 Visualização

Nesta fase desenvolvemos uma interface para visualização dos dados na forma geográfica. Permitindo assim, estudar a correlação dos dados. As consultas são realizadas de modo nativo no banco de dados MySQL.

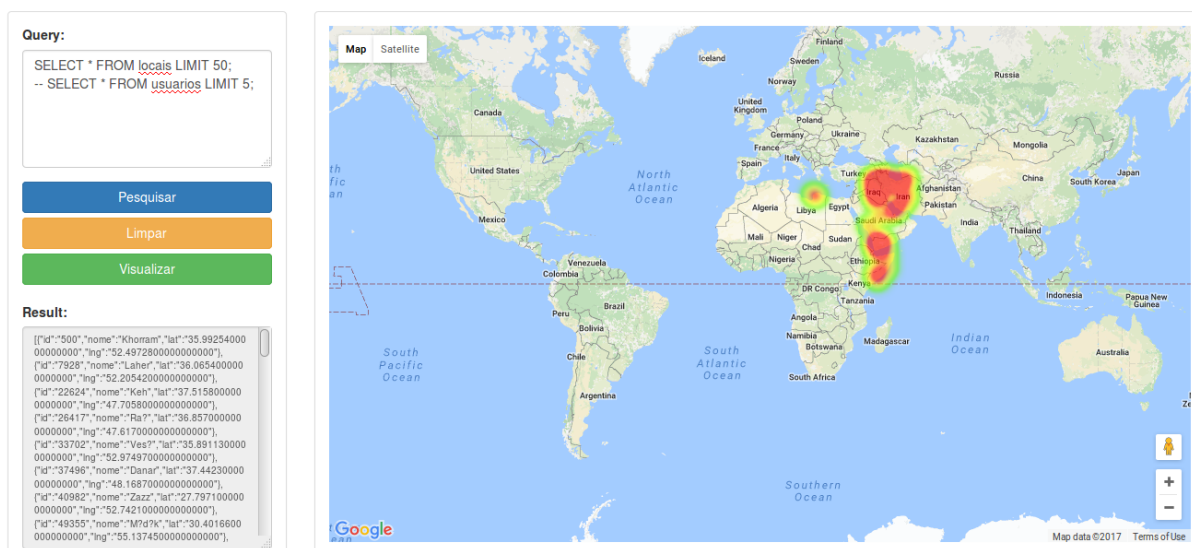


Figura 10 – Interface para Visualização das Informações

4 Análises e Resultados

De acordo com as imagens a seguir, percebemos que as redes de compartilhamento de conhecimento estudadas neste trabalho, estão presentes em todo o mundo. Possuindo uma pequena concentração na China, Índia e Europa.

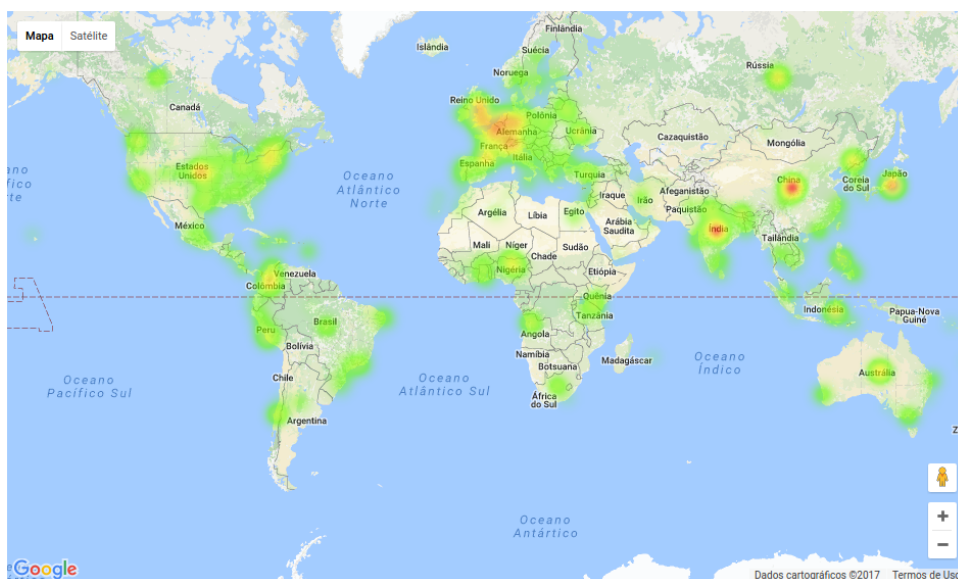


Figura 11 – Usuários das Redes

Verificando apenas o Git Hub, identificamos uma maior utilização na China e Europa.

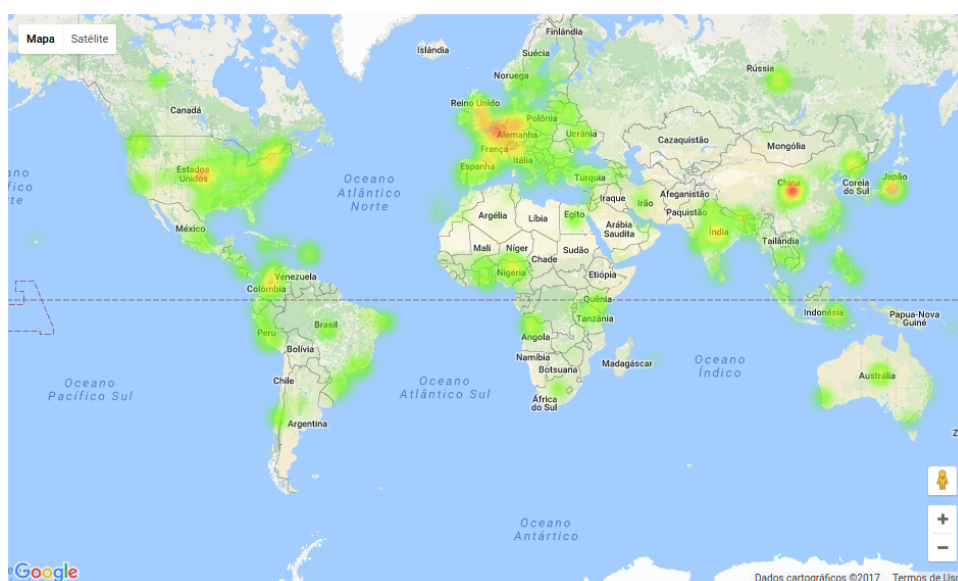


Figura 12 – Usuários Git Hub

Já o Stack Overflow, é utilizado predominantemente na Índia e Europa.

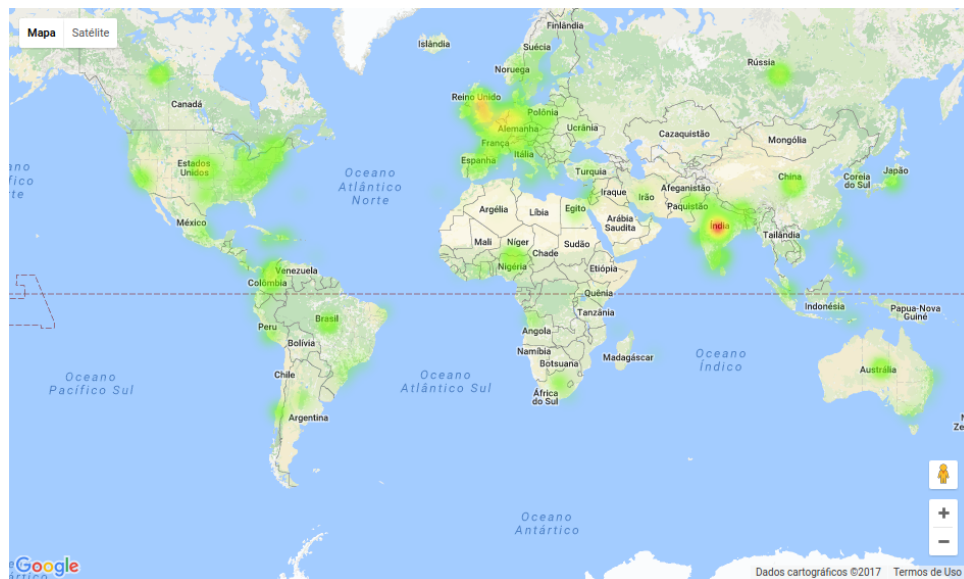


Figura 13 – Usuários Stack Overflow

Identificamos também que os usuários dos Estados Unidos e Inglaterra são os que mais informam apenas a sigla do País.

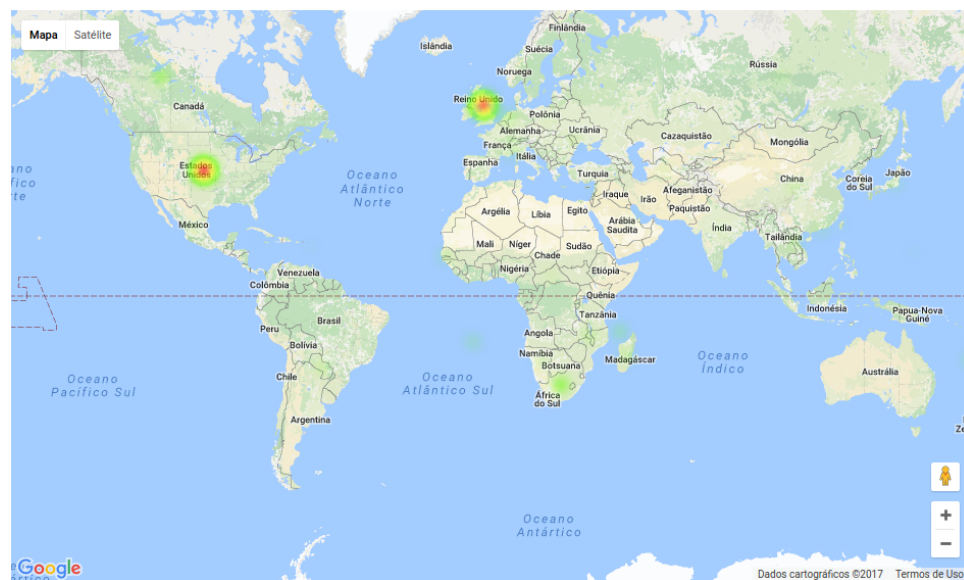


Figura 14 – Resultado: Algoritmo R0

As localizações encontradas pelo algoritmo R1 são mais heterogêneas e bem distribuídas pelo mundo.

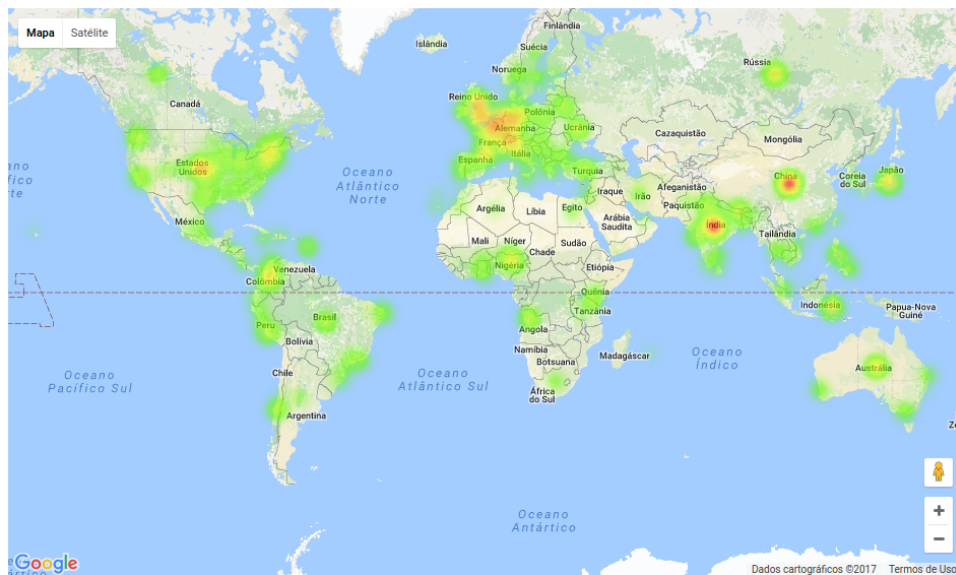


Figura 15 – Resultado: Algoritmo R1

O algoritmo R2 teve uma concentração maior na China devido a muitos dos seus endereços serem escritos em Mandarim.



Figura 16 – Resultado: Algoritmo R2

Podemos verificar também que os usuários da América Central e Europa são os que mais erram a digitação dos endereços.

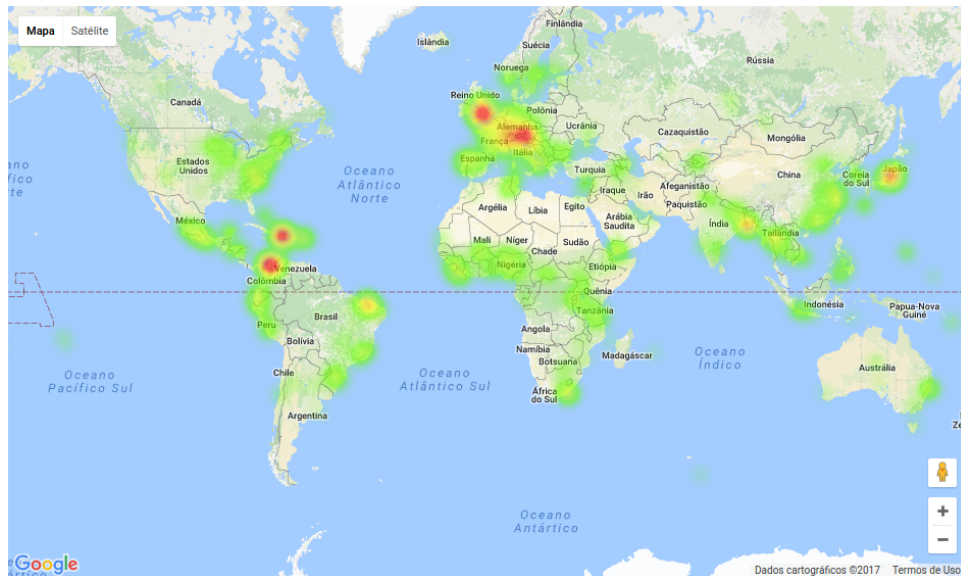


Figura 17 – Resultado: Algoritmo R3

Nossa última análise trata das palavras que não possuem um local fixo. Aqui, identificamos que poucos são os nomes dos lugares.



Figura 18 – Nuvem de Palavras sem Localização

5 Conclusão

O presente trabalho foi completo no sentido em que trouxe consigo diversos desafios, desde a aplicação de conhecimentos obtidos em outras disciplinas como: Mineração de Dados, Banco de Dados Avançados(NoSQL), Processamento de Linguagem Natural, Algoritmo e Estrutura de Dados(Paralelismo e Programação Dinâmica), Banco de Dados Geográficos e Armazém de Dados. Além de integrar diversas tecnologias(Shell Script, RegEx, Python, PHP, MongoDB e MySQL) e interfaces de programação de aplicativos(APIs - Google Maps).

Outra oportunidade se fez no processo de geocodificação, onde testamos novas ideias para identificar as coordenadas a partir de endereços informados de modo livre. Fator de grande importância uma vez que este tipo de endereço representa grande parte dos dados na internet hoje.

Para trabalhos futuros, recomendamos que seja implementado uma comunicação com as APIs, bancos de dados externos e ou serviços de consulta em nuvem disponibilizados pelos sites aqui relacionados. Desse modo, podemos ampliar a capacidade de análise das informações.

Referências

BADASHIAN, A. S. et al. Involvement, contribution and influence in github and stack overflow. In: IBM CORP. *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering*. [S.l.], 2014. p. 19–33.

HUANG, W. et al. Cpdscorer: Modeling and evaluating developer programming ability across software communities.

JR, C. A. D.; FONSECA, F. T. Assessing the certainty of locations produced by an address geocoding system. *Geoinformatica*, Springer, v. 11, n. 1, p. 103–129, 2007.

MARTINS, D.; JR, C. A. D.; FONSECA, F. T. Geocodificação de endereços urbanos com indicação de qualidade. *Proceedings XIII GEOINFO*, p. 36–41, 2012.

NAVARRO, G. A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, ACM, v. 33, n. 1, p. 31–88, 2001.

VASILESCU, B.; FILKOV, V.; SEREBRENIK, A. Stackoverflow and github: Associations between software development and crowdsourced knowledge. In: IEEE. *Social Computing (SocialCom), 2013 International Conference on*. [S.l.], 2013. p. 188–195.

WIKIPEDIA. *GitHub — Wikipedia, The Free Encyclopedia*. 2016. [Online; accessed 9-September-2016]. Disponível em: <https://en.wikipedia.org/w/index.php?title=GitHub&oldid=738492841>.

WIKIPEDIA. *Stack Overflow — Wikipedia, The Free Encyclopedia*. 2016. [Online; accessed 9-September-2016]. Disponível em: https://en.wikipedia.org/w/index.php?title=Stack_Overflow&oldid=736163294.

WU, S.; MANBER, U. Fast text searching: allowing errors. *Communications of the ACM*, ACM, v. 35, n. 10, p. 83–91, 1992.