Léxico para Recursos Naturais e Meio Ambiente

Danilo Yudi Futata Kassuya
Departamento de Computação
Universidade Estadual de Londrina
Londrina-Paraná, Brasil
danilo.yudi.futata@uel.br

Felipe Alves Barusso
Departamento de Computação
Universidade Estadual de Londrina
Londrina-Paraná, Brasil
felipe.barusso@uel.br

Guilherme Henrique G. Silva
Departamento de Computação
Universidade Estadual de Londrina
Londrina-Paraná, Brasil
guilherme.henrique.silva@uel.br

Cinthyan Renata Sachs Camerlengo de Barbosa
Departamento de Computação
Universidade Estadual de Londrina
Londrina-Paraná, Brasil
cinthyan@uel.br

Wagner Ferreira Lima
Departamento de Letras Vernáculas e Clássicas
Universidade Estadual de Londrina
Londrina-Paraná, Brasil
wflima@uel.br

ABSTRACT

This paper intended to create a lexicon, where the sequences of tokens are representations of the basic vocabulary of Natural Resources and the Environment. It has the entries that are considered the most relevant in environmental studies and they were selected among the most frequently used ones. In order to create the lexicon, a hash table was implemented, where each item has two parts: the key and the object. The key is the word itself to be searched in this lexicon, and the object contains the information about this word. It was possible to observe that the creation of this lexicon allows a search for natural resources and environment concepts with good computational performance.

KEYWORDS

Lexicon, Natural Resources, Environment, Hash Table.

1 INTRODUÇÃO

Praticar a sustentabilidade é pensar no futuro. Assim, vários desenvolvedores têm criado aplicativos para celulares e tablets que incentivam o comportamento sustentável e a preservação do meio ambiente como uma forma de reduzir os impactos ambientais causados pelos maus hábitos de consumo ou pela falta da conscientização.

Lima *et al.* [1] fizeram um levantamento no período de 2016 a 2020 de aplicativos móveis voltados a temas ambientais, indicando que houve uma expansão no número desses, em destaque para categorias ligadas à Gestão Ambiental e Educação Ambiental, bem como se evidenciou uma quantidade média maior de *downloads* pelos usuários nas categorias de sustentabilidade e biodiversidade, respectivamente, demonstrando assim que a demanda por aplicativos na área ambiental está em crescimento diante da necessidade de soluções objetivas e acessivas à população.

Independentemente dos números apresentarem uma preferência dos desenvolvedores pela criação de aplicativos voltados à gestão e educação ambiental, com nicho de usuários específicos, a preferência dos usuários no ato de baixar tais tecnologias tem sido por aplicativos inerentes à sustentabilidade e biodiversidade, respectivamente [1].

Porém, percebe-se a falta de aplicativos no formato de dicionários *online* na área de Recursos Naturais ou Meio Ambiente. O que temos achado na web são livros escaneados sobre assuntos específicos como Glossário de Termos Técnicos Ambientais Rodoviários do DNIT (Departamento Nacional de Infra-Estrutura de Transportes) [2], onde não conseguimos fazer uma busca rápida geral sobre termos do Meio Ambiente e Recursos Naturais. Encontram-se ainda alguns pequenos dicionários disponíveis de iniciativa de outros órgãos, como por exemplo, o Dicionário de Termos e Expressões Técnicas relativas ao Direito Ambiental que a procuradoria do Ministério Público (MP) do Mato Grosso criou com 37 termos [3]. O MP do Amapá está reproduzindo esse material e consta apenas com 34 termos [4].

O objetivo deste trabalho foi desenvolver um léxico digital no domínio de Recursos Naturais e Meio Ambiente baseado em um dicionário disponível do Instituto Brasileiro de Geografia e Estatística (IBGE) [5]. A escolha desse léxico foi feita também por essa área ter diversas palavras complexas e específicas para o público em geral e possuir uma diversidade grande de palavras que seria interessante estar em um único sistema de Processamento de Linguagem Natural (PLN), o qual foi implementado com uma função hash com desempenho ótimo disponível em Moreno [6].

2 DESENVOLVIMENTO DO LÉXICO

Para o desenvolvimento deste Léxico foi preciso não só estudar o domínio de Recursos Naturais e do Meio Ambiente, mas também a utilização de funções de busca hash [6] que tenha um bom desempenho. Parte do dicionário (tem 1382 palavras) pode ser vista na Figura 1.

Há grandes esforços na construção de dicionários e léxicos computacionais para PLN [7]. Assim, uma alternativa para o bom espalhamento dessas palavras seria o uso da Tabela Hash [6]. A referida estrutura de dados é também conhecida por outras

denominações como *tabela esparsa*, *tabela de dispersão*, *tabela de espalhamento* ou simplesmente *tabela hashing*.

```
from hash_table import inserir


def criar_dicionario(tabela_hash):
    inserir(tabela_hash, ['abalo', 'Vibração do solo devido a um sismo (terremoto) ou explosão'])

    inserir(tabela_hash, ['acaule', 'Denominação aplicada a uma planta que não apresenta caule visível.'])

    inserir(tabela_hash, ['acetato', 'Sal derivado do ácido acético, sendo em geral um sólido cristalino.'])

    inserir(tabela_hash, ['acrófita', 'Planta que vive nas regiões alpinas.'])

    inserir(tabela_hash, ['acrografia', 'Arte de gravar em relevo, através da utilização da água-forte.'])

    inserir(tabela_hash, ['ácron', 'Parte anterior não segmentada do corpo de um animal metamérico.'])
```

Figura 1: Léxico de Recursos Naturais e Meio Ambiente [Os autores]

Tabelas *Hash* são um tipo de estruturação para o armazenamento de informação de forma extremamente simples, fácil de implementar e intuitiva quando se trata de organizar grandes quantidades de dados. O conceito central é a divisão de um universo de dados a ser organizado em subconjuntos mais facilmente gerenciáveis. A estruturação da informação em tabelas *hash* visa principalmente permitir armazenar e procurar rapidamente grande quantidade de dados [8].

3 METODOLOGIA

Para a criação do léxico foi implementada uma tabela hash indicada por Moreno [6] e Moreno, Barbosa e Manfio [7]. Essa estrutura de dados funciona por meio da associação de chaves de busca com valores. Assim, ao utilizar uma palavra do dicionário como chave, podemos realizar uma busca sobre suas propriedades com baixo custo computacional.

Variações na técnica de pesquisa, conhecidas como *hashing*, têm sido implementadas em muitos sistemas. O conceito de *Hashing aberto* refere-se à propriedade de que não precisa haver limite no número de entradas que podem ser feitas em uma tabela. Esse esquema nos dá ainda a capacidade de realizar *e* entradas sobre *n* nomes num tempo proporcional a $n(n+e)/m$, para qualquer constante *m* de nossa escolha [9].

É importante ressaltar que, para realizar a inserção de cada valor distinto a um índice único, seria necessária uma função hash perfeita. No algoritmo desenvolvido (bem como na maioria dos casos) foi escolhida uma função hash imperfeita, o que significa que pode haver colisão. Uma colisão é o evento de dois objetos distintos serem atribuídos o mesmo índice no vetor associativo da tabela. Para contornar esse problema é necessário implementar soluções que garantam o funcionamento da tabela no caso de índices iguais serem atribuídos a objetos distintos.

Tais soluções consistem em percorrer os objetos de cada índice do vetor associativo. Assim, chega-se em uma complexidade de pior caso de O(n) para operações de inserção, busca e remoção. O algoritmo desenvolvido cria a tabela hash, insere o dicionário nela e em seguida permite ao usuário realizar consultas.

Na etapa de inserção, cada palavra é enviada, junto de suas propriedades, à função denominada "inserir". Nela, o programa calcula o índice em que essa palavra deve ser inserida em um vetor por meio de uma função denominada "função_hash".

O algoritmo com a função *Hash One at a Time* desenvolvido por Jenkins [10] é mais simples em relação à função Mix [6], sendo que o cálculo ocorre para cada caractere da chave utilizando os operadores "<<" (shift-left) e ">>"(shift-right) para deslocamento de bits. Os passos seguintes podem ser vistos em Moreno [6]:

a) realiza para cada caractere da chave o seguinte cálculo:

$$hash+ = Chave[i];\ hash+ = (hash \ll 10);\ hash \wedge = (hash \gg 6);$$

b) após as operações com todos os caracteres são realizadas as últimas operações binárias :

$$hash+ = (hash \ll 3);\ hash \wedge = (hash \gg 11);\ hash+ = (hash \ll 15);$$

c) para adequar o valor do *hash* ao tamanho da tabela, a seguinte instrução é realizada:

$$hash\ \&\ TamanhoTabela$$

Nesse léxico em particular foi adotada tal função. Ela realiza uma série de operações *bitwise* para cada letra da chave (nesse caso um termo do dicionário) e retorna um índice. Os autores optaram por esse método, pois ele foi desenvolvido para trabalhar com strings (textos).

One at a Time se mostrou ser uma função que obteve os melhores resultados em outros léxicos [6] e a sugestão é que trabalha melhor com tamanhos de tabelas partindo de 4% da dimensão do léxico.

Porém, não é possível garantir que o índice gerado seja único para todas as chaves. Por isso, empregamos o conceito de *buckets*. Cada posição do vetor é uma lista e, caso duas palavras sejam inseridas no mesmo índice, o programa anexa o termo novo no final da lista encadeada. A posição da tabela aponta para o primeiro elemento na lista e, se não houver elementos, a posição é NIL [8].

Para determinar se existe uma entrada para uma cadeia de caracteres s na tabela de símbolos, aplicamos uma *função de hash h* a *s* de tal forma que *h*(*s*) retorne um inteiro entre 0 e m-1. Se *s* estiver na tabela de símbolos, estará na lista enumerada por *h*(*s*) e se ainda não estiver é introduzida por meio da criação de um registro para a mesma, que é ligado ao início da lista numerada por *h*(*s*). A lista média tem um comprimento de *n*/*m* registros, se existirem *n* nomes numa tabela de comprimento *m* [9].

Por fim, com a flexibilidade da linguagem Python foi possível inserir objetos complexos no vetor da tabela hash com facilidade. Assim, optou-se por escolher inserir uma tupla formada pela própria palavra (chave) e seus atributos como mostra a Figura 2.

```
inserir(tabela_hash, ['abalo', 'Vibração
do solo devido a um sismo (terremoto) ou
explosão'])
```

Figura 2: Inserção de uma tupla formada pela palavra (utilizada como chave) e sua definição [Os autores]

A experiência do usuário é feita por meio de um menu, onde ele pode inserir uma palavra. Se essa palavra for "sair", o programa é finalizado. Caso contrário, é realizada uma busca (por um processo de calcular o índice similar à etapa de inserção) e as propriedades daquela palavra são exibidas pelo programa.

4 RESULTADOS

Com uma tabela de tamanho 2048, o programa inseriu 1382 termos do dicionário que envolvem Recursos Naturais e Meio Ambiente, em 1003 índices distintos. Além disso, o índice com mais objetos teve cinco palavras inseridas.

A média do tempo de execução na etapa de inserção de 1382 termos no dicionário foi de 6,0010 milissegundos, como é mostrada na Figura 3.

```
Tempo de insercao:  6001000 ns
Digite 'sair' para fechar o programa
Entrada:
```

Figura 3: Tempo de inserção de 1382 objetos [Os autores]

Já a média do tempo de execução da etapa de busca foi complexa de obter, devido ao intervalo minúsculo de tempo que uma busca leva. Assim, foi desenvolvido um teste, baseado em Moreno [7], para executar 1000 buscas 100 vezes e calcular a média de tempo de 1000 buscas. O resultado obtido foi 0.20006 milissegundos para cada 1000 buscas (Figura 4), demonstrando a eficiência da tabela hash com a função escolhida.

```
Digite 'sair' para fechar o programa
Entrada: debug
Media de tempo de 1000 buscas (10 execucoes): 200060.0
Entrada:
```

Figura 4: Tempo de execução de 1000 buscas [Os autores]

Todos os cálculos de tempo de execução foram realizados utilizando a biblioteca *time* da linguagem Python. No início de cada execução, foi obtido o tempo do sistema. Após o fim da execução, o processo foi repetido. O tempo de execução foi obtido com a diferença entre os dois valores, como indica a Figura 5.

```
start_time = time.time_ns()
        for j in range(1000):
            busca = buscar(tabela_hash, 'abalo')
        total = time.time_ns() - start_time
```

Figura 5:Exemplo de obtenção de tempo de execução[Os autores]

5 CONCLUSÕES

A análise da criação de um léxico, cujo domínio é do Meio Ambiente e Recursos Naturais, foi escolhida neste trabalho. Um sistema de PLN desenvolvido em Python foi desenvolvido para validar uma função hash para a criação de chaves que ligam aos seus atributos que contém informações do referido domínio.

Todo esse processo nos forneceu uma grande visão sobre ampliar o leque de abrangência desse vocabulário relacionado ao Meio Ambiente e difundir conceitos de forma ágil, facilitando pesquisas por vezes demoradas.

Assim, a função hash aqui utilizada (*One at a Time*) foi uma solução adequada para implementação do Léxico para Recursos Naturais e ao Meio Ambiente.

Sabe-se que com o crescimento pela demanda de aplicativos em consonância com o desenvolvimento tecnológico e social, isso exige soluções cada vez mais rápidas e eficazes diante de problemas decorrentes do aumento da escala de produção e consumo desenfreado da atividade humana.

A inovação tecnológica se apresenta como uma alternativa para concretizar a busca por soluções para a crise ambiental. O desenvolvimento tecnológico é um aliado da conservação ambiental.

Este trabalho tem ainda a vantagem de poder ser aplicado em outra área de conhecimento, bastando para isso inserir um novo léxico, tendo assim um grande impacto na aplicação geral dessa tecnologia. Isso é possível devido à flexibilidade do algoritmo utilizado. Já no domínio deste trabalho que é Recursos Naturais e o Meio Ambiente, outras fases de análises, como Sintática e Semântica, poderiam ser expandidas.

REFERÊNCIAS

[1] A. Z. S. Lima, C. R. O. Carneiro, L. G. Furtado, V. A. Batista e A. N. Pontes. 2020. Tecnologia e meio ambiente: levantamento de aplicativos móveis voltados a temas ambientais. In *Brazilian Journal of Development.* 6, 9 (Sep., 2020), 68090-68105. DOI: https://doi.org/10.34117/bjdv.v6i9

[2] DNIT- Departamento Nacional de Infra-Estrutura de Transportes. Diretoria de Planejamento e Pesquisa. Coordenação Geral de Estudos e Pesquisa. Instituto de Pesquisas Rodoviárias. 2006. *Glossário de termos técnicos rodoviários*. Rio de Janeiro. https://www.gov.br/infraestrutura/pt-br/centrais-de-conteudo/glossario_ tecnicos_ambientaisterrestrednit10-08-06.pdf.

[3] MP-MT – Ministério Público. 2017. *Ação: Orientação Técnica para Valoração de Danos Ambientais. Dicionário de Termos e Expressões Técnicas*. https://pjedaou.mpmt.mp.br/wp-content/uploads/2017/10/Dicion%C3%A1rio-de-Termos-1.pdf

[4] MP-AP – Ministério Público. Centro de Apoio Operacional do Meio Ambiente. 2019. *Dicionário de Termos e Expressões Técnicas relativas ao Direito Ambiental*. https://www.mpap.mp.br/caop-meio-ambiente? view=article& id=7759:dicionario-de-termos-e-expressoes-tecnicas-relativas-ao-direito-ambiental &catid=142

[5] IBGE – Instituto Brasileiro de Geografia e Estatística. 2004. *Vocabulário Básico de Recursos Naturais e Meio Ambiente.* Rio de Janeiro: IBGE.

[6] Fábio C. Moreno. 2017. *Visual Tahs: Ferramenta para analisar a eficácia de buscas das funções hash em um léxico para língua natural*. Londrina: Departamento de Computação da Universidade Estadual de Londrina. Dissertação de Mestrado.

[7] Fábio C. Moreno, Cinthyan R. S. C. de Barbosa e Edio R. Manfio. 2021. Tabelas Hash para um Léxico Digital. *Revista de Informática Teórica e Aplicada* (*RITA*), 28, 2 (Ago., 2021), 26-38. DOI: https://doi.org/10.22456/2175-2745.107128

[8] Thomas H. Cormen. Algoritmos: Teoria e Prática. Rio de Janeiro: Campus, 2002. v.2.

[9] Alfred V. Aho, Ravi Sethi and Jeffrey D. Ullmann. 1995. *Compiladores:* Princípios, técnicas e ferramentas. Trad. Daniel A. Pinto. Rio de Janeiro: LTC - Livros Técnicos e Científicos Editora SA.

[10] Bob Jenkins. 1997. *Algorithm Alley*. https://www.drdobbs.com/database/algorithm-alley/184410284