Um Estudo Avaliativo de Algoritmos de *Searching*

Jonas M. Chagas[1]

[1]Campus Alegrete – Universidade Federal do Pampa
Avenida Tiarajú, 810 – Alegrete, RS, Brazil

jonaschagas10@gmail.com

Abstract. *Nowadays, access to information is becoming easier due to the Internet. When there is a need to find or to know about something, a search is executed on sites such as Google. Research algorithms have a fundamental role in computing, and are increasingly evolving to supply the users needs. Thus, in the computer literature there are several search algorithms, some traditional and others being tested all the time. Thus, it is proposed in this work to perform a evaluative study of 2 traditional search algorithms, in order to validate the behaviors described in the literature. The evaluation will have a bias of 3 metrics: data set input form, algorithm complexity and execution time. As a result, after the execution of the algorithms it was found that the complexity analysis described in the literature is very reliable with the behavior of the running algorithm. It was also possible to analyze that the format of the data set directly influences the algorithms performance.*

Resumo. *Nos dias atuais, o acesso a informação está cada vez mais facilitado devido a internet. Quando há a necessidade de encontrar ou saber sobre algum assunto é realizado uma pesquisa em sites como Google. Algoritmos de pesquisa possuem um papel fundamental na computação, e estão evoluindo, cada vez mais, para suprir as necessidades dos usuários. Assim, na literatura da computação existem diversos algoritmos de pesquisa, alguns mais tradicionais e outros propondo novas abordagens. Com isso, propõe-se neste trabalho realizar um estudo avaliativo de 2 algoritmos tradicionais de pesquisa, afim de validar os comportamentos descritos na literatura. A avaliação terá o viés de 3 métricas: forma da entrada do conjunto de dados, complexidade do algoritmo e tempo de execução. Como resultados, após a execução dos algoritmos foi constatado que a análise de complexidade descrita na literatura é bem fidedigna com o comportamento do algoritmo em execução. Também foi possível analisar que a formatação do conjunto de dados influência diretamente no desempenho dos algoritmos.*

1. Introdução

Nos dia atuais, o acesso a informação está cada vez mais facilitado devido a internet e os motores de pesquisa. Toda vez que é preciso buscar alguma informação na rede normalmente é realizado uma pesquisa em sites como por exemplo o *Google*. Os algortimos de pesquisa possuem um papel fundamental no mundo da computação. Então, uma pesquisa é realizada com sucesso, se dada a entrada de uma chave/valor a saída seja correspondente à essa entrada. Na literatura existem vários métodos de pesquisa, como pesquisa em largura, pesquisa em profundidade, pesquisa linear, entre outros. Esses algoritmos possuem suas particularidades, pontos negativos e positivos.

Com isso, propõe-se neste trabalho um estudo avaliativo de 2 (dois) algoritmos de pesquisa retirados da literatura que serão implementados em uma lista simples. Será realizado a avaliação através de 3 métricas: tempo de execução do algoritmo, complexidade do algoritmo e forma da entrada do conjunto de dados. Após a execução dos experimentos comprovou-se que a análise da complexidade dos algoritmos descritas na literatura é bem fidedigna com a execução dos algoritmos. A pesquisa binária possui tempo de execução constante, embora aumente o tamanho da lista de elementos, a pesquisa possui a mesma performance. Já a pesquisa linear possui tempo de execução linear, uma vez que a lista de elementos aumenta, o tempo de execução aumenta.

O restante do trabalho está organizado como segue. Na seção 2 são apresentados os referenciais teóricos dos algoritmos utilizados neste trabalho. Na seção 3 é descrito o desenvolvimento do estudo e implementações dos algoritmos. Na seção 4 é reportada a metodologia da avaliação, e os resultados obtido ao final. Por fim, na seção 5 são apresentadas as conclusões e trabalhos futuros.

2. Referencial Teórico

Nesta seção são abordados os conceitos teóricos dos algoritmos de pesquisa linear e pesquisa binária.

2.1. Algoritmo de Pesquisa Linear

O algoritmo de pesquisa linear é bem eficiente e pode ser implementado em diversas estruturas de dados. Esta pesquisa tem como pior caso O(n), ou seja, no pior o caso a pesquisa itera sobre todos elementos até encontrar a chave/valor. Já o melhor caso da pesquisa linear acontece quando a chave/valor encontra-se no inicio da lista, tendo complexidade O(1), pois itera somente em um elemento [Skiena 2008]. A implementação do algoritmo é bem simples e de fácil entendimento [Skiena 2008], abaixo um pseudocódigo da pesquisa linear.

Algorithm 1 Pesquisa Linear

```
1: função PESQUISA LINEAR(A[], chave)
2:     para i ← 1 até A[n] faça
3:         se A[i] = chave então
4:             devolve i
5:         fim se
6:     fim para
7:     devolve Não se encontra "chave"em A[]
8: fim função
```

Dado uma lista de elementos a pesquisa ocorre a partir do primeiro elemento da lista que é comparado com a chave/valor de busca, caso não seja o elemento correspondente a pesquisa continua através dos elementos n vezes até ser encontrado.

Segundo Skiena existem alguns fatores que devem ser analisados antes da implementação. Uma delas é que a pesquisa linear é de facil implementação, então se o programador não possuir tempo para projetar outros algoritmos mais complexos é uma boa escolha [Skiena 2008].

2.2. Algoritmo de Pesquisa Binária

A pesquisa binária possui uma complexidade maior na implementação que a pesquisa linear. É um algoritmo que utiliza a metodologia de divisão e conquista, ou seja, divide o problema em partes menores para sua resolução, após a resolução de todas partes, é preciso unir todas as partes para solução do problema maior. A complexidade no pior caso da pesquisa binária é de O(log n), que é uma grande solução [Skiena 2008]. A implementação do algoritmo pode ocorrer de duas formas: recursiva ou iterativa. Abaixo um exemplo de implementação na forma recursiva.

Algorithm 2 Pesquisa Binária

```
    função PESQUISABINARIA(A[], chave, esquerda, direita)
        meio;
2:      se esquerda > direita então
            devolve -1
4:      fim semeio = (esquerda + direita) / 2;
        se A[meio] = chave então
6:          devolve meio;
        fim se
8:      se A[meio] > chave então
            devolve (PESQUISABINARIA(A[], chave, esquerda, meio - 1));
10:     senão
            devolve (PESQUISABINARIA(A[], chave, meio + 1, direita));
12:     fim se
    fim função
```

O algoritmo recebe quatro parâmetros, uma lista de dados ordenados, chave/valor de pesquisa e os limites inferior e superior da lista. Primeiramente é testado um dos casos base. Após é atribuido a uma variável o valor da metade da lista, assim é testado se o valor da metade da lista corresponde a chave/valor. Se for verdadeiro o algoritmo encerra, se for falso segue para o próximo teste. Se o valor da metade da lista for maior que o valor da chave/valor a pesquisa segue para o lado esquerdo da lista, se for ao contrário transcorre para o lado direito.

Algumas questões devem ser respondidas antes da implementação do algoritmo, tais como:

- A lista de dados estará sempre ordenada?
- Qual tamanho possui a lista de dados?
- Qual estrutura de dados será utilizada?

A primeira pergunta é muito importante, pois a pesquisa binária só funciona em dados ordenados, então se os dados não estiverem ordenados deve-se levar em consideração o tempo de ordenação antes da pesquisa. Quanto a segunda questão, para uma lista de dados relativamente pequena deve-se levar em consideração a pesquisa linear que para poucos elementos possui boa eficiência [Skiena 2008]. A última questão leva em consideração que a pesquisa binária não é possivel ser implementada em algumas estruturas, pois ela precisa do *index* da estrutura.

3. Desenvolvimento do Trabalho

Na subseção 3.1 é apresentado o problema e as discussões sobre o mesmo, e na subseção 3.2 é descrito a configuração do ambiente que foi executado os experimentos.

3.1. Problema Proposto: Algoritmos de Pesquisa

Problema da Pesquisa: Dado um conjunto de dados 'S', onde se encontra a chave 'q' em 'S'?

Existem muitos algoritmos de pesquisa, alguns provavelmente muito complexos que devem ser utilizados em estruturas mais complexas ainda, porém neste trabalho serão abordados dois tradicionais algoritmos, a pesquisa sequencial e a pesquisa binária.

3.1.1. Discussões do Problema

Segundo Skiena, antes de escolher qual algoritmo de pesquisa será implementado, é preciso responder algumas questões:

1. Quanto tempo você pode gastar em programação?
2. Alguns itens são acessados com mais frequência do que outros?
3. As frequências de acesso podem mudar com o tempo?
4. A chave está por perto?
5. Minha estrutura de dados está na memória externa?
6. Posso adivinhar onde deve estar a chave?

A primeira pergunta é a principal questão à ser respondida, uma vez que se haver tempo é possível investir em um algoritmo mais sofisticado e mais complexo, analisando também se há a necessidade do programa ter um algoritmo desses. Então para a primeira questão é preciso levar em consideração a complexidade do conjunto de dados e também o tempo disposto para implementação.

A segunda questão tem relação com realizar uma otimização em algum algoritmo tradicional [Skiena 2008], uma vez que se é possível ter a informação sobre quais palavras repetidas são acessadas constantamente é possível implementar algum algoritmo para que essas palavras fiquem à frente das demais. Porém, segundo o Skiena algumas empresas trabalham com o acesso uniforme das informações, ou seja, as palavras que menos são acessadas devem ter a mesma chance probabilística que as demais.

Já a terceira questão tem relação com a frequencia de acesso aos dados terá o programa. Uma vez que o projetista possui essa informação tem a possibilidade de escolher o algoritmo que será compatível e a estratégia de como organizar esses dados na estrutura de dados escolhida. Assim, é possível adiantar dados que são pesquisados, trocando eles de posição com os dados da frente sob demanda, sem precisar ter um controle [Skiena 2008].

A quarta questão tem o viés que se possuir a informação de onde está a chave da pesquisa é possível encontrá-la utilizando algoritmos tradicionais em um tempo de execução satisfatório. Também é possível utilizar saltos em uma pesquisa sequencial, testar exponencialmente os indices da estrutura de dados [Skiena 2008].

A quinta questão se refere a quantidade de chaves, primeiramente se o conjunto de dados é grande, o algoritmo a ser escolhido tende a ser a pesquisa binária, contudo se for muito grande ela pode extrapolar a memória saltando a cada ponto médio em busca da chave [Skiena 2008].

Por último, a sexta questão está relacionada com um algoritmo de pesquisa que pode saltar em busca da chave, Skiena faz uma analogia a uma pesquisa em uma lista telefonica, que é possível achar a fatia da lista onde esta a chave da pesquisa na primeira comparação, ao invés de 2 pelo menos que a pesquisa binária realiza.

É possível realizar uma reflexão sobre as perguntas de maneira que todas informações que são coletadas anteriormente são importantes, contudo algumas não parecem ser de fato algo praticado no mundo real, como a última questão, pois qualquer mudança na estrutura de dados já interfere no algoritmo implementado e o trabalho é perdido.

3.2. Configuração do Ambiente

A tabela 1 descreve o ambiente utilizado nos experimentos.

Tabela 1. Configuração do Ambiente de Execução

Notebook	*Vaio*	
Processador	Processador Intel Core i5 - 7200 2.70GHz	
Memória RAM	8GB	
Sistema Operacional	*Windows* 10 64 bits	
Linguagem de programação	*Python*	
IDE	*PyCharm Community Edition*	
Bibliotecas	Sorteio da chave/valor	*Random*
	Captura do tempo	*TimeIt*
	Gráficos	*MatPlotLib*

4. Avaliação dos Experimentos

Nesta seção são abordados os conteúdos relacionados aos experimentos realizados e as decisões tomadas.Na subseção 4.1 é descrito a metodologia de avaliação. Por fim, na subseção 4.2 é demonstrado os resultados dos experimentos e as comparações realizadas entre os algoritmos.

4.1. Metodologia da Avaliação dos Experimentos

A avaliação dos experimentos ocorreu através de três métricas: entrada de dados, tempo de execução e complexidade do algoritmo.

A Tabela 2 apresenta as entradas de dados definidas para as execuções dos experimentos.

A definição das entradas levou em consideração tanto dados ordenados quanto desordenados. Já que o algoritmo de pesquisa binária só possui êxito em dados ordenados essa execução terá o acréscimo de uma ordenação antes de efetuar a pesquisa binária. Para pesquisa linear não influência os dados estarem ordenados ou não.

Tabela 2. Entrada de Dados para Execuções dos Experimentos

Número de Elementos	Ordenados	Desordenados
1.000	x	x
4.000	x	x
5.000	x	x
6.000	x	x
10.000	x	x

Para a realização da avaliação do tempo de execução dos algoritmos foi preciso utilizar uma biblioteca chamada *timeit* do *Python* que auxiliou na coleta do tempo de execução. Foram capturados os tempos de 5 execuções de cada pesquisa para cada entrada de dados. Através desses dados foi possível calcular a média aritmética e o desvio padrão de cada entrada de dados.

Já a complexidade do algoritmo foi calculada através das funções:

1. **Função linear:** é calculada através da fórmula: f(x) = (a * x) + b;
2. **Função logarítmica:** é calculada através da fórmula: f(x) = $\log_x * a$;

A complexidade foi utilizada para realizar uma comparação do tempo de execução dos algoritmos nos experimentos com o cálculo realizado pelas funções descritas.

4.2. Avaliação dos Algoritmos

Após as execuções dos experimentos foram gerados 2 gráficos de avaliação dos algoritmos, 1 gráfico com dados de entrada ordenados e outro com dados desordenados. A figura 1 apresenta as avaliações relacionadas com dados de entrada ordenados.

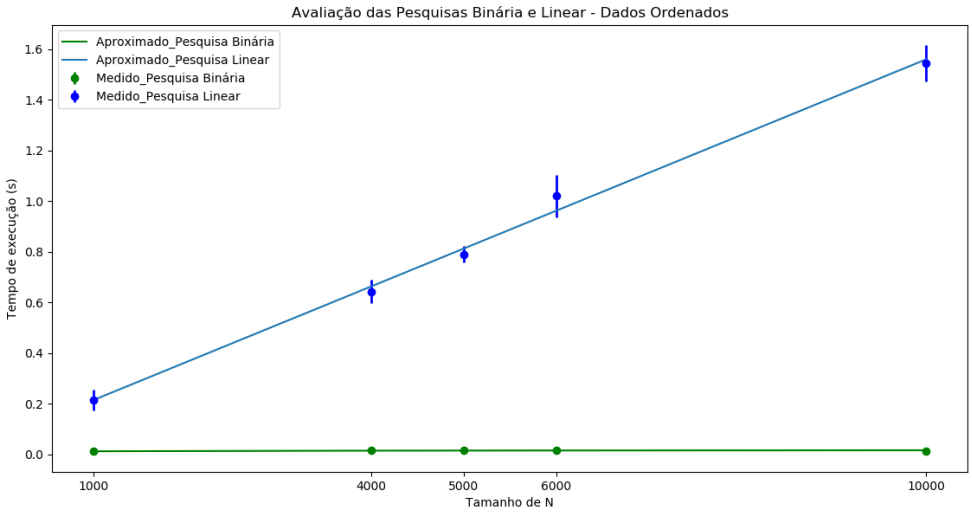

Figura 1. Avaliação dos Algoritmos de Pesquisa Linear e Binária - Conjunto de Dados Ordenados

No gráfico é possível identificar 2 linhas contínuas e 10 pontos. As linhas contínuas representam os cálculos realizados pelas funções linear e algorítmica. Já os pontos representam as coletas dos tempos de execução dos algoritmos nos experimentos.

Primeiramente, é possível verificar que as avaliações relacionadas a complexidade dos algoritmos corroboram com a análise de complexidade do livro do Skiena, uma vez que os pontos coletados das execuções estão bem próximos das linhas contínuas, ou seja, o desempenho dos algoritmos condizem com a análise de complexidade descrita na literatura do Skiena.

A entrada de dados ordenados não afetou o desempenho de nenhum algoritmo, uma vez que ambos funcionam com dados ordenados. Com isso, é possível verificar que o tempo de execução da pesquisa binária é constante, ou seja, embora o número de elementos aumente a pesquisa segue com a mesma perfomance. Já a pesquisa linear possui o tempo linear, já que conforme o número de elementos aumenta o tempo de execução da pesquisa aumenta também.

Também é possível realizar um comparativo sobre a performance entre os dois algoritmos e é notável que a pesquisa binária possui uma perfomance melhor que a pesquisa linear, já que o tempo de execução é bem abaixo da pesquisa linear.

Já a figura 2 demonstra as avaliações realizadas para dados desordenados. Nessa avaliação os dados desordenados possuem um papel importante, uma vez que a pesquisa binária só funciona com dados ordenados. Diante disso, foi preciso realizar uma ordenação antes de efetuar a pesquisa binária, e esse acréscimo foi acrescentado tanto na execução do algoritmos de pesquisa binária quanto no cálculo da função logarítmica. A pesquisa linear não afetou pois funciona para dados ordenados ou desordenados.

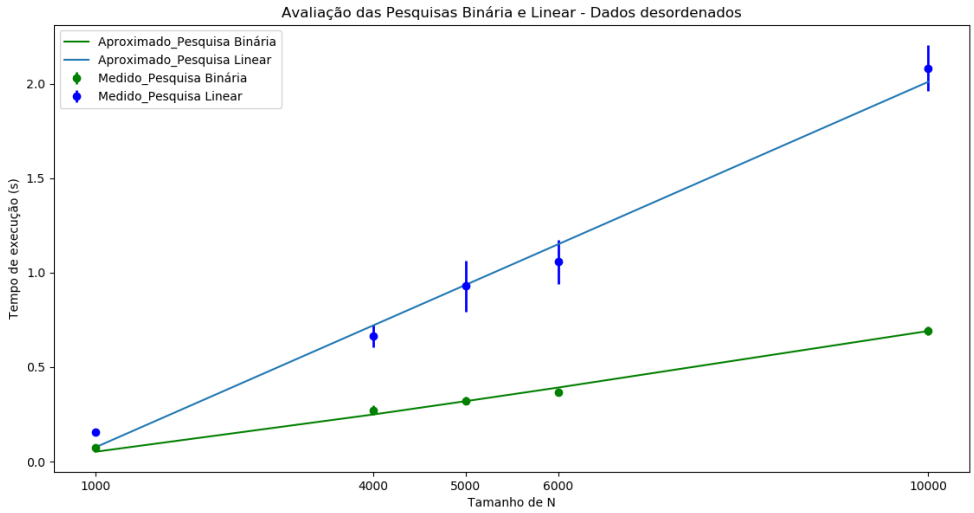

Figura 2. Avaliação dos Algoritmos de Pesquisa Linear e Binária - Conjunto de Dados Aleatórios

Através do gráfico é possível verificar que a avaliação da complexidade do algoritmo de pesquisa linear não modificou pois os pontos coletados estão próximos da linha contínua calculada pela função linear. A avaliação da complexidade da pesquisa binária também comprovou que os pontos coletados em execução coincidem com a linha contínua calculada através da função logarítmica e da função de ordenação.

O tempo de execução do algoritmo de pesquisa binária aumentou relacionando

o mesmo com o gráfico 1, pois foi preciso realizar a ordenação dos dados para então a pesquisa binária ser chamada. Contudo, os tempos demonstrados nos gráficos mostram que nesse experimento é evidente que a pesquisa binária possui melhor performance que a pesquisa linear, até acrescentando o tempo da ordenação.

5. Conclusão

O objetivo deste trabalho foi realizar um estudo avaliativo de dois algoritmos de pesquisa. Após a execução dos experimentos e coleta de dados, os gráficos gerados foram bem conclusivos em duas questões. Primeiramente a análise da complexidade do algoritmo descritas pela literatura do Skiena são bem fidedignas com o comportamento do algoritmo em execução. Segundo, sempre antes de implementar algum algoritmo, tanto de pesquisa como de qualquer outro propósito deve-se analisar todo o contexto em que ele será submetido.

Também foi possível verificar a eficiência do algoritmo de pesquisa binária quando recebe um conjunto de dados ordenados, a diferença é grande em relação ao algoritmo de pesquisa linear. Com dados aleatórios, para a realização da pesquisa binária é preciso ordenar os dados se tornando um algoritmo com tempo de execução um pouco maior. Já a pesquisa linear não possui problemas relacionados com a entrada de dados, porém possui tempo de execução linear, ou seja, a partir que a lista de dados aumenta, o tempo de execução aumenta. Também é possível realizar um comparativo entre os dois algoritmos, sendo possível observar que a pesquisa binária possui melhor performance que a pesquisa linear.

Para trabalhos futuros é possível realizar uma avaliação dos algoritmos em estrutura de dados diferentes, uma vez que somente foi utilizado a lista contígua. Também é possível coletar mais métricas além do tempo de execução, como por exemplo a memória do computador, uma vez que a pesquisa binária possui uma implmentação recursiva, que pode ocasionar um estouro da memória. Realizar uma pesquisa mais profunda relacionada a literatura para verificar o que outros autores pensam sobre o problema do algoritmo de pesquisa.

Referências

Skiena, S. S. (2008). *The algorithm design manual: Text*, volume 2. Springer Science & Business Media.