Análise de Complexidade e Experimental de Algoritmos de Ordenação

Bruno Santos de Lima
Faculdade de Ciências e Tecnologia
Universidade Estadual Paulista
Presidente Prudente, Brasil
brunoslima@gmail.com

Leandro Ungari Cayres
Faculdade de Ciências e Tecnologia
Universidade Estadual Paulista
Presidente Prudente, Brasil
leandroungari@gmail.com

***Resumo*—O**

Palavras-chave: ordenação, análise assintótica e análise experimental.

I. Introdução

Diversas aplicações da atualidade envolvem um grande volume de dados, desde aplicações comerciais simples a grande aplicações científicas, todas estão nesse contexto. A organização estrutural de conjunto de dados, além de prover melhor usabilidade, também otimiza tempo e o consumo de recursos, tanto de processamento quanto de memória para a execução.

Neste contexto, este trabalho apresenta uma análise dos principais algoritmos de ordenação, o qual está dividido nas seguintes seções: na Seção 2, são apresentados os algoritmos de ordenação utilizados, indicando a abordagem utilizada no processo juntamente com a sua respectiva análise assintótica. Na Seção 3, são apresentados os estudos comparativos analisando a variações dos conjunto de dados de entrada e abordagens utilizadas na ordenação. Por fim, na Seção 4, são apresentadas as considerações finais do estudo.

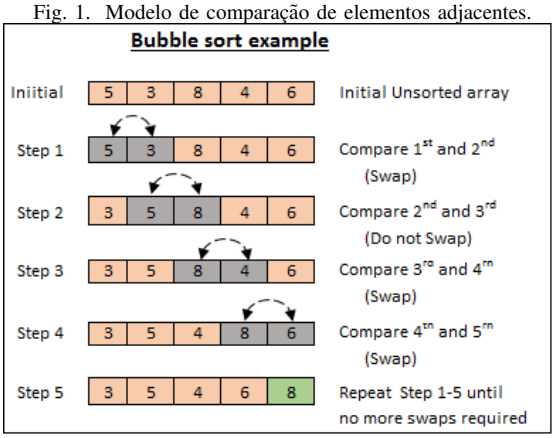
II. Algoritmos de Ordenação

A. Algoritmos baseados em Troca

1) Bubble Sort: O presente algoritmo utilizada a abordagem de troca, através da permutação de elementos vizinhos, seguindo a ideia de densidade dos elementos, em que pode-se optar por varrer o arranjo levando o maior elemento (mais pesado) ao fim do vetor, ou conduzir o menor elemento (mais leve) ao início desse. Esse passo (uma das duas opções excludentemente) é realizado sucessivamente para os subvetores não ordenados remanescentes até que o vetor esteja ordenado como um todo.

A abordagem original prevê que todos os elementos adjacentes sejam comparados através de $n-1$ iterações, porém é possível que o arranjo esteja ordenado sem que sejam necessárias todos esses ciclos. De modo a prevenir operações desnecessárias, algumas abordagens utilizam-se de *flags* para a detecção de trocas, caso a última iteração tenha ao menos uma ocorrência, há necessidade de pelo menos mais uma iteração, em caso negativo, o processo pode ser encerrado.

Fig. 1. Modelo de comparação de elementos adjacentes.



A Figura 1 apresenta a abordagem de ordenação utilizada pelo algoritmo referido.

Complexidade:

- **Melhor caso:** $O(n)$, para vetor crescente somente na implementação com melhoria.
- **Caso médio:** $O(n^2)$.
- **Pior caso:** $O(n^2)$, para vetor decrescente e aleatório.

2) Quick Sort: O algoritmo foi proposto por C.A.R. Hoare em 1962, tem como estratégia a divisão do arranjo original em partições a partir da determinação de um pivô. Para qualquer abordagem de escolha do pivô, em cada partição busca-se encontrar os elementos maiores que o pivô a partir do início e os menores a partir do fim do arranjo, os quais são trocados de posição aos pares, de modo a ordenar a partição, se necessário também através da quebra de subpartições.

Em relação a abordagem de escolha de pivô, há diversas técnicas que buscam explorar possíveis características dos elementos do arranjo. Dentre as principais, pode-se destacar a escolha do elemento inicial ou final do vetor, o uso de propriedades estatísticas como a média e mediana, entre outras. O principal objetivo de qualquer uma dessas estratégias é minimizar a ocorrência dos piores casos de recorrência, resultando em um comportamento assintótico quadrático.

Geralmente, os piores caso desse algoritmo consistem na escolha de pivô como elemento mínimo ou máximo, para entradas de dados crescentes e decrescentes.

Complexidade:

- **Melhor caso:** $O(nlogn)$.
- **Caso médio:** $O(nlogn)$.
- **Pior caso:** $O(n^2)$, em geral para o pivô mínimo e máximo.

B. Algoritmos baseados em Inserção

1) Insertion Sort: A estratégia de ordenação por inserção consiste em um dos métodos mais simples, sendo extremamente eficiente em conjuntos pequenos, com estratégia de percorrer o esquerdo da esquerda para a direita deixando os elementos ordenados à esquerda. Uma situação cotidiana que aplica a abordagem referida é a inserção de cartas na mão de um jogador, seguindo a estrutura de dados deque.

Complexidade:

- **Melhor caso:** $O(n)$, para arranjo crescente.
- **Caso médio:** $O(n^2)$.
- **Pior caso:** $O(n^2)$.

2) Shell Sort: O algoritmo Shell Sort foi proposto por Donald Shell em 1959, consistindo em um dos métodos de ordenação mais eficientes dentre os modelos quadráticos, baseando-se no Insertion Sort, a partir de comparações com elementos não adjacentes, desse modo, facilitando o deslocamento dos menores elementos para o início do vetor através do uso de saltos regressivos de tamanho.

O grande diferencial desse método é a utilização dos saltos, porém não existe uma abordagem única para o cálculo do tamanho dos saltos, consistindo em um fator determinante na mensuração da complexidade assintótica. Dentre os principais modelos têm-se o n primeiros múltiplos de um fator ou combinação de fatores (ex. 2^p3^q) ou os n primeiros números primos.

A Figura 2 apresenta a ordenação das partições de elementos distantes, com saltos de tamanho 40, 13 e 4 respectivamente, partindo do arranjo inicial e alcançando o conjunto ordenado.

Complexidade:

- **Caso médio:** $O(n^{3/2})$

C. Algoritmos baseados em Seleção

1) Selection Sort: O algoritmo Selection Sort, trata-se modelo mais básico de algoritmo de ordenação, utilizada muitas em ambientes naturais e é mais intuitivo para os seres humanos, pois se utiliza da abordagem de força bruta, sem

Fig. 2. Modelo de partições com elementos distantes em diferentes tamanhos.

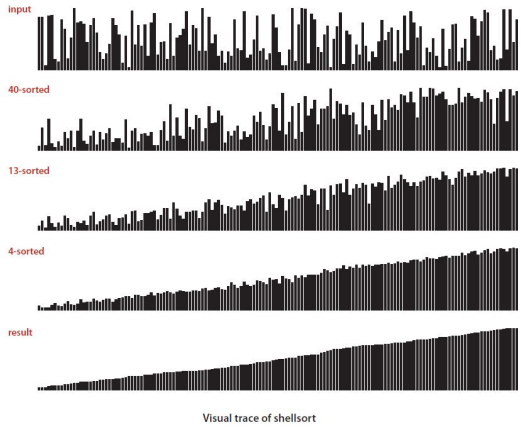
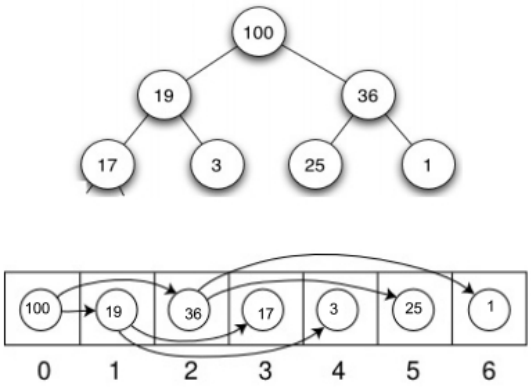


Fig. 3. Equivalência entre a árvore e arranjo.



aproveitar nenhuma peculiaridade do conjunto de dados ou vantagem que possa ser tomada. Sua estratégia consiste em percorrer todo o arranjo n, comparando todos os elementos de forma a encontrar o menor, em seguida, o segundo menor e assim por diante, de modo a organizar todo o conjunto.

O grande destaque para esse algoritmo, dentro do contexto das abordagens quadráticas, consiste no reduzido número de trocas, que no máximo ocorre n vezes.

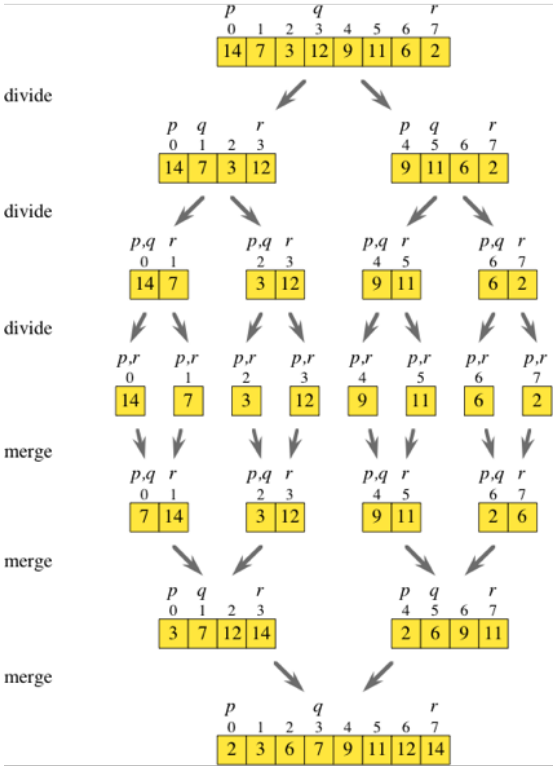
Complexidade:

- **Caso médio:** $O(n^2)$, de modo invariante.

2) Heap Sort: Consiste em um algoritmo de ordenação extremamente sofisticado cuja implementação baseia-se em uma fila de prioridades utilizando a estrutura de dados Heap, o que consiste em uma árvore binária em vetor. Foi inventado por J.W.J. Williams em 1964. O algoritmo consiste em construir um max heap, em seguida trocar o primeiro elemento com o último, e por fim rearranjar o max heap, para todos os elementos exceto para o último, que já é o maior elemento do vetor [2].

A Figura 3 apresenta o processo de equivalência entre a estrutura de dados Heap na forma de árvore e arranjo.

Fig. 4. Abordagem de divisão e conquista do algoritmo.



Complexidade:

- **Caso médio:** $O(n^2)$, de modo invariante.

D. Algoritmos baseados em Intercalação

1) Merge Sort: A Figura 4 apresenta o processo de divisão e conquista utilizado no algoritmo.

Complexidade:

- **Caso médio:** $O(n^2)$, de modo invariante.

III. Análise Experimental

Após a realização da análise assintótica, foi realizada a análise experimental dos mesmos algoritmos, tais testes envolveram diferentes tamanhos de entrada, as dimensões utilizadas para o vetor foram: 10, 100, 1 000, 10 000, 100 000 e 1 000 0000 de elementos, em todos os casos citados acima foram utilizados como teste um vetor crescente, decrescente e elementos dispersos pelo vetor de forma aleatória, ressaltando que para este último foi utilizado um arquivo que armazena os elementos, para que esta sequência aleatória seja sempre a mesma para execução em todos os algoritmos, para todos os testes o tempo é dado em milissegundos (ms).

Para a realização dos testes foi utilizado um computador com as seguintes especificações: processador Intel Core i3 - 3110M CPU @ 2.40 GHz, memória RAM 4 GB, memória caches L1 de 128 KB, L2 de 512 KB e L3 de 3 MB.

A. Modelos de entrada de dados

1) Entrada crescente:
2) Entrada decrescente:
3) Entrada aleatória:

B. Comparativo de abordagem

1) Abordagem de inserção:
2) Abordagem de seleção:
3) Abordagem de troca:

IV. Considerações Finais

Referências

[1] Programa das Nações Unidas para o Desenvolvimento (PNUD). *Atlas do desenvolvimento humano do Brasil*. PNUD; 2003. Disponível em: http://www.pnud.org.br/atlas/

[2] Sen AK. *Desenvolvimento como liberdade*. São Paulo: Companhia das Letras; 2000.