Capítulo 6

Sobre algoritmos de ordenação e sua abordagem no ensino médio

Me. Ilso Francisco dos Santos[1]
Dr. Marcelo Pedro dos Santos[2]

Resumo: O presente artigo se baseia no trabalho *Algoritmos de Ordenação: uma abordagem didática para o ensino médio* (SANTOS, 2020). A proposta do trabalho se concentra numa abordagem didática de três algoritmos de ordenação tomando como base competências e habilidades propostas na Base Nacional Curricular Comum (BNCC). Os algoritmos de ordenação estudados foram o *Bubble Sort*, *Selection Sort* e o *Quick Sort*. Inicialmente foi realizada uma análise de competências e habilidades presentes não somente na BNCC, o documento mais recente, mas também nos Parâmetros Curriculares Nacionais de Matemática: 5ª a 8ª séries, do Ensino Médio, Complemento para o Ensino Médio e dos Parâmetros Curriculares do Estado de Pernambuco. Os conteúdos didáticos necessários para a compreensão do trabalho são: Somatórios e Funções Afim, Quadrática e Logarítmica, necessárias para avaliarmos os desempenhos assintóticos dos algoritmos. Alguns

[1]Professor da rede pública Estadual e Municipal de Ensino em Pernambuco, ilsofrancisco@gmail.com.

[2]Professor do Departamento de Matemática - Universidade Federal Rural de Pernambuco, marcelo.pedrosantos@ufrpe.br.

exemplos dos algoritmos servirão para o entendimento do funcionamento, bem como a forma como sua complexidade é analisada e propostas de atividades são colocadas no intuito de mostrar a viabilidade da abordagem do tema no Ensino Médio.

Palavras-chave: Parâmetros Curriculares; Competências; Algoritmos de Ordenação; Complexidade.

6.1 Introdução

De acordo com as Diretrizes Curriculares Nacionais da Educação Básica, a escola, no desempenho de suas funções, deve construir e utilizar métodos, estratégias e recursos de ensino que melhor atendam às características cognitivas e culturais dos alunos. O tema Algoritmos de Ordenação propicia uma abordagem diversificada da Matemática em um contexto que auxilia no desenvolvimento cognitivo e cultural dos estudantes. O presente trabalho se propõe a analisar, à luz da BNCC, o funcionamento e a complexidade dos algoritmos de ordenação *Bubble Sort*, *Selection Sort* e *Quick Sort*, comparando seus desempenhos por meio de funções matemáticas, tanto algebricamente quanto graficamente.

Apresentaremos exemplos dos algoritmos de ordenação e atividades afins com o propósito de trazer subsídios educacionais para uma abordagem didática do tema no Ensino Médio. Apesar da análise dos Parâmetros Curriculares se estenderem ao Ensino Fundamental, optou-se por direcionar o trabalho com o tema para o Ensino Médio. A justificativa de tal direcionamento deve-se ao fato de se supor que os estudantes já possuam um repertório matemático capaz de compreender os conceitos e procedimentos abordados no estudo dos algoritmos já citados. O tema dos algoritmos está intrinsecamente relacionado à computação, no entanto não faremos tal abordagem neste trabalho e nos resumiremos ao viés matemático do tema, recorrendo de forma superficial aos conteúdos de Somatórios e Funções Afim, Quadrática e Logarítmica.

Tendo como fundamento que o Ensino Médio é uma etapa de aprofun-

damento de conceitos e procedimentos trabalhados com os estudantes no Ensino Fundamental, consideramos plausível uma análise dos Parâmetros Curriculares nessa etapa do Ensino. Não se pretende aprofundar a matemática utilizada para obter a Função que representa a complexidade de cada algoritmo e nem a análise assintótica de cada função. A proposta é apenas mostrar, de forma inteligível, como se pode chegar aos resultados e a maneira que tais resultados são utilizados na análise dos algoritmos. Para tanto, traremos exemplos de cada algoritmo de ordenação, com suas respectivas funções representativas e complexidades correlatas. No que diz respeito a análise assintótica, utilizada para representar a complexidade de cada algoritmo, nos limitaremos a apresentar qual a representação da mesma sem que haja preocupação de como se chegou a tal representação, pois tais demonstrações fogem do escopo do trabalho.

Por fim, deve-se atentar para o fato de que o estudo do tema não se encerra nessa simples abordagem, restando um vasto campo a ser investigado por outros pesquisadores com focos diversos dos que foram aqui destacados.

6.2 Fundamentos teóricos e metodológicos

6.2.1 Análise dos parâmetros curriculares

Inicialmente buscou-se mostrar uma possível evolução curricular nos documentos oficiais, ou melhor, nos Parâmetros Curriculares Nacionais e também do Estado de Pernambuco, na busca de competências e habilidades que tivessem relação com o tema. Os Parâmetros Curriculares Nacionais de Matemática 5ª a 8ª séries (PCN - 5ª a 8ª) já trazem uma problematização ao declararem:

> Situação - problema é o ponto de partida da atividade matemática e não a definição. No processo de ensino e aprendizagem, conceitos, ideias e métodos matemáticos devem ser abordados mediante a exploração de problemas, ou seja, de situações em

> que os alunos precisem desenvolver algum tipo de estratégia para resolvê-las (BRASIL, 1998, p. 40).

Segundo o mesmo documento, é por meio da exploração de situações-problemas que o estudante reconhecerá diferentes funções da álgebra, bem como exercitará a indução e a dedução em Matemática. Apesar dos PCN - 5ª a 8ª séries já sugerirem o quão importante é uso da tecnologia, do computador e de softwares no processo de ensino - aprendizagem, ainda não se faz menção ao tema algoritmos. Nos *Parâmetros Curriculares Nacionais de Matemática*, afirma-se que: "muitos conteúdos importantes são descartados por serem julgados, sem uma análise adequada, que não são de interesse para os alunos porque não fazem parte de sua realidade ou não têm uma aplicação prática imediata" (BRASIL, 1998). Encontramos também nesse documento uma referência aos conteúdos procedimentais e a capacidade do saber fazer. É evidente que o trabalho com algoritmos se enquadra em tal espécie de conteúdo, visto que se trata de um procedimento ou conjunto deles.

Na etapa do Ensino Médio, deve-se propor ao estudante uma ampliação dos conhecimentos adquiridos no Ensino Fundamental, bem como uma elevação no nível de abstração de conceitos e procedimentos matemáticos. De acordo com os Parâmetros Curriculares Nacionais para o Ensino Médio (PCNEM), essa etapa "[$\cdots$] deve ser mais do que memorizar resultados dessa ciência e que a aquisição do conhecimento matemático deve estar vinculada ao domínio de um saber fazer Matemática e de um saber pensar matemático"(BRASIL, 2000). Também há de se garantir, nessa etapa do ensino, que o estudante seja capaz de lidar com o conceito de funções em situações diversas, ajustando seus conhecimentos na busca de soluções de problemas, na construção de modelos para interpretação e investigação em Matemática. Também é preciso garantir que ele aprofunde conhecimentos sobre números e álgebra, de forma conectada com outros conteúdos. Nos PCNEM, algumas competências de base foram elencadas em duplas, as de representação e comunicação, bem como as de investigação e compreensão. A relação de tais competências com o tema Algoritmos de Ordenação é

bastante clara, pois a análise destes proporciona a representação e a comunicação por meio de esquemas, linguagem natural e fluxogramas, assim como instiga a investigação e a compreensão quando se busca assimilar o procedimento utilizado por cada algoritmo e a função que descreve o número de comparações necessárias para ordenar a lista (vetor), como será visto adiante. Também se averiguou, na versão complementar para o Ensino Médio, os PCNEM$_+$, um enfoque na contextualização dos conteúdos matemáticos e na interdisciplinaridade, retomando as competências elencadas nos PCNEM, dentre as quais podemos destacar:

> Traduzir uma situação dada em determinada linguagem para outra; por exemplo, transformar situações dadas em linguagem discursiva em esquemas, tabelas, gráficos, desenhos, fórmulas ou equações matemáticas e vice-versa, assim como transformar as linguagens mais específicas umas nas outras, como tabelas em gráficos ou equações (BRASIL, 2002, p. 115).

A competência citada anteriormente está inserida nas de representação e comunicação, que possuem ampla relação com a proposta didática em tela, pois o trabalho com Algoritmos de ordenação favorece toda essa gama de representações e formas de comunicar o conhecimento matemático a partir da situação-problema. Os PCNEM$_+$ também propõem o ensino por meio de temas estruturadores para que se alcancem as competências e habilidades desejadas. Nesse caso, o tema Algoritmos de Ordenação poderia servir como estruturador, pois favorece a articulação lógica entre diferentes ideias e conceitos matemáticos.

Já os Parâmetros Curriculares da Educação Básica do Estado de Pernambuco também trazem tópicos semelhantes aos apresentados anteriormente nos documentos já citados, como a resolução de problemas e a modelagem matemática, mas difere na nomenclatura dada às competências e habilidades. Nesse documento, usa-se a expressão Expectativas de Aprendizagem:

> As expectativas de aprendizagem explicitam aquele mínimo que o estudante deve aprender para desenvolver as competências básicas na disciplina. Em outras palavras, elas descrevem

> o "piso" de aprendizagens, e não o "teto". Dependendo das condições de cada sala de aula, elas podem ser ampliadas e/ou aprofundadas (PERNAMBUCO, 2012, p. 13).

O documento destaca, dentre outras coisas, o trabalho com situações-problemas, os problemas abertos em detrimento dos fechados e a modelagem matemática. Algumas expectativas de aprendizagem que possuem relação com o estudo de Algoritmos de Ordenação perpassam por todo os ensino médio como "reconhecer função como modelo matemático para o estudo das variações entre grandezas do mundo natural ou social" (PERNAMBUCO, 2012, p. 21).

A BNCC, último e mais recente documento analisado, traz competências e habilidades relacionadas com o trabalho sobre Algoritmos de Ordenação, tanto na parte destinada para Ensino Fundamental, quanto para o Ensino Médio. Como o aprofundamento e ampliação dos conceitos devem ser consolidados no Ensino Médio, esse documento propõem o desenvolvimento de competências e habilidades de forma análoga aos PCNEM. De acordo com a BNCC: "Competência é definida como a mobilização de conhecimentos (conceitos e procedimentos), habilidades (práticas cognitivas e socioemocionais), atitudes e valores para resolver demandas complexas da vida [$\cdots$]" (BRASIL, 2018). Na seção do Ensino Fundamental, são explicitadas competências gerais da Educação Básica e competências específicas de cada área do conhecimento, dentre as quais são indicadas as habilidades requeridas. Os conteúdos, por outro lado, são separados em unidades temáticas do 1º ao 9º ano, como vem a seguir, para a disciplina de Matemática: Números, Álgebra, Geometria, Grandezas e Medidas, Probabilidade e Estatística. Consta neste documento que "A linguagem algorítmica tem pontos em comum com a linguagem algébrica, sobretudo em relação ao conceito de variável" (BRASIL, 2018, p. 271). Quanto às habilidades diretamente relacionadas ao tema, destacamos:

> -Representar por meio de um fluxograma os passos utilizados para resolver um grupo de problemas.

> -Descrever, por escrito e por meio de um fluxograma, um algoritmo para a construção de um triângulo qualquer, conhecidas as medidas dos três lados.
>
> -Descrever, por escrito e por meio de um fluxograma, um algoritmo para a construção de um polígono regular (como quadrado e triângulo equilátero), conhecida a medida de seu lado.
>
> -Identificar a regularidade de uma sequência numérica ou figural não recursiva e construir um algoritmo por meio de um fluxograma que permita indicar os números ou as figuras seguintes (BRASIL, 2018).

Percebe-se um uso pretensioso de fluxogramas em várias habilidades mas não há, ainda, relação com o pensamento computacional que "envolve as capacidades de compreender, analisar, definir, modelar, resolver, comparar e automatizar problemas e suas soluções, de forma metódica e sistemática, por meio do desenvolvimento de algoritmos" (BRASIL, 2018, p. 474).

A estruturação das competências e habilidades para o Ensino Médio se fez, na BNCC, de forma distinta do Ensino Fundamental. No Ensino Médio, as competências específicas não estão diretamente relacionadas à unidades temáticas, configurando uma liberdade de associação de acordo com a situação-problema, tema estruturador ou itinerários formativos que sejam escolhidos. Porém, merecem destaque algumas habilidades que possuem forte relação com o tema Algoritmos de Ordenação. São elas:

> -Investigar e registrar, por meio de um fluxograma, quando possível, um algoritmo que resolve um problema.

> -Utilizar conceitos iniciais de uma linguagem de programação na implementação de algoritmos escritos em linguagem corrente e/ou matemática (BRASIL, 2018, p. 538-539).

Observa-se que no Ensino Médio já se busca uma relação mais estreita com o pensamento computacional ao se referir à linguagem de programação e implementação de algoritmos, fato não antes visto na parte do Ensino Fundamental.

Por fim, pudemos constatar que são notáveis as competências e habilidades constantes nos Parâmetros Curriculares analisados que podem ser desenvolvidas por meio da abordagem didática do tema proposto neste trabalho. E, como a BNCC é o documento mais recente, faz-se necessário mais estudos e propostas tangíveis ao processo de ensino-aprendizagem na educação básica.

6.2.2 Algoritmos de ordenação e complexidade

Um Algoritmo de Ordenação tem como procedimento ordenar os n elementos de uma lista finita, a qual passaremos a chamar de vetor, em ordem crescente ou decrescente.

Os primeiros contatos que os estudantes têm com o significado de algoritmo se dá com o estudo de procedimentos matemáticos, mas sabemos ser tal definição mais abrangente. Segundo Ribeiro, algoritmo é "um conjunto de regras e operações bem definidas e ordenadas, destinadas à solução de um problema ou de uma classe de problemas, em um número finito de etapas" (2018, p. 32). Já para Cormen et al., "um algoritmo é qualquer procedimento computacional bem definido que toma algum valor ou um conjunto de valores como **entrada** e produz algum valor ou conjunto de valores como **saída**" (2002). Este último acrescenta que problemas de ordenação são comuns e que a ordenação é uma operação fundamental na ciência da computação.

Uma ordenação por **comparação simples** realiza ou não a troca dos elementos dos vetores dois a dois após compará-los como é caso dos algo-

ritmos ***Bubble Sort*** e ***Selection Sort***. Já há os algoritmos que utilizam uma ordenação por comparação mais complexa chamada de **método eficiente** como ocorre com o ***Quick Sort***. Este último possui uma técnica chamada **divisão e conquista** que consiste em dividir um problema a ser resolvido em problemas menores, que podem ser divididos em partes ainda menores. A partir disso, encontram-se as soluções dos problemas menores e combina-se as soluções dos problemas menores em uma solução global, que resolve o problema inicial.

Para realizar-se a comparação da eficiência dos Algoritmos de Ordenação utilizamos a **notação assintótica** ou notação **O-grande**, representada pelo símbolo O. que se refere a um limite assintótico superior. Vale salientar que, ao analisar assintoticamente os Algoritmos de Ordenação, considera-se o termo de maior crescimento da lei de formação da função representante do desempenho do Algoritmo de Ordenação. Entretanto, não faz parte do escopo deste trabalho um aprofundamento, tanto no que diz respeito à representação da notação assintótica acima citada, quanto do motivo da escolha do termo de maior crescimento que nos referimos anteriormente.

> Quando observamos tamanhos de entrada grandes o suficiente para tornar relevante apenas a ordem de crescimento do tempo de execução, estamos estudando a **eficiência assintótica** dos algoritmos. Ou seja, estamos preocupados com a maneira como o tempo de execução de um algoritmo aumenta com o tamanho da entrada no *limite*, à medida que o tamanho da entrada aumenta indefinidamente. Em geral, um algoritmo que é assintoticamente mais eficiente será a melhor escolha para todas as entradas, exceto as muito pequenas (CORMEN et al., 2002, p. 51).

Para que se tenha mais clareza desse conceito segue uma definição da notação **O-grande** e um simples exemplo, pois tal notação será utilizada mais adiante.

Para uma dada função $g(n)$, denotamos por $O(g(n))$ o conjunto de funções $O(g(n)) = \{f(n)$: *existem constantes* c e n_o *tais que* $0 \leq f(n) \leq c \cdot g(n)$ *para todo* $n \geq n_o\}$.

Exemplo 1. *Considere as funções* $f(n) = n^2 + 5n + 1$ *e* $g(n) = n^2$.
Tem-se, para $n \geq 1$, $n^2 + 5n + 1 \leq n^2 + 5n^2 + n^2 = 7 \cdot n^2 \Longrightarrow f(n) \leq 7 \cdot g(n)$.
Portanto:

$$f(n) \in O(g(n)).$$

Segundo Cormen et al., "para indicar que uma função $f(n)$ é membro de $O(g(n))$, escrevemos $f(n) = O(g(n))$" (2002, p. 35).

No caso do vetor já estar ordenado configura-se o **melhor caso** da ordenação e, se estiver em ordem contrária, teremos o **pior caso da ordenação**. Existe ainda o **caso médio** da ordenação que considera o tempo médio para se percorrer $n/2$ elementos do vetor até obter o elemento procurado na ordenação. Contudo, o caso médio difere dependendo do tipo de algoritmo, sendo objeto de um estudo mais aprofundado que não é o foco deste trabalho.

Apesar do conceito de Fluxogramas estar associado ao estudo de algoritmos, merecendo um destaque nas competências e habilidades em matemática na BNCC, não faremos tal abordagem neste documento e sugerimos uma leitura do capítulo 3 de Santos (2020).

6.2.2.1 *Bubble Sort*

O Algoritmo de Ordenação ***Bubble Sort*** é considerado um dos mais simples. É também conhecido como Ordenação por Flutuação, daí o nome ***Bubble***, cuja tradução para o Português é **Bolha**. Seu procedimento se dá da seguinte maneira: para uma ordenação crescente, iniciando-se com o primeiro elemento, compara-se cada elemento do vetor com o posterior, caso este seja maior, muda-se de posição com o anterior, caso não o seja, a troca de posição não é feita e passa-se para o próximo par de comparação até que não haja mais comparações e o vetor fique ordenado. Para uma ordenação decrescente, compara-se o elemento com o posterior, caso este

seja maior, não se muda sua posição atual. Segue-se para o próximo par de comparação e assim sucessivamente até que o vetor esteja ordenado. Além da simplicidade, outra vantagem desse Algoritmo de Ordenação é a estabilidade, mas a lentidão o deixa em desvantagem com relação a outros Algoritmos de Ordenação. É indicando para pequenos vetores e contraindicado para vetores com um número grande de elementos.

Tomemos, como exemplo, o vetor $A = (5,4,3,2,1)$, que possui o número de elementos $n = 5$. O objetivo é colocá-lo em ordem crescente com a estratégia de comparação do *Bubble Sort*. O vetor A caracteriza o pior caso do *Bubble Sort*.

Os procedimentos se dão da seguinte maneira:
Passo I. 5 compara com 4 e troca de posição, pois 5 > 4;
Passo II. 5 compara com 3 e troca de posição, pois 5 > 3;
Passo III. 5 compara com 2 e troca de posição, pois 5 > 2;
Passo IV. 5 compara com 1 e troca de posição, pois 5 > 1.
Agora, ocorre o mesmo com o 4.
Passo V. 4 compara com 3 e troca de posição, pois 4 > 3;
Passo VI. 4 compara com 2 e troca de posição, pois 4 > 2;
Passo VII. 4 compara com 1 e troca de posição, pois 4 > 1.
Passo VIII. 3 compara com 2, como 3 < 2, troca de posição com este.
Passo IX. 3 compara com 1 e também troca de posição com ele.
Passo X. 2 compara com 1 e troca de posição.

Outra forma de representar o procedimento está descrito na tabela a seguir:

Tabela 6.1: Exemplo do *Bubble Sort* com 5 elementos, $n = 5$, representando o pior caso.

VETOR	**Nº de comparações**
5-4-3-2-1	4 comparações (5 e 4, 5 e 3, 5 e 2, 5 e 1).
4-3-2-1-5	3 comparações (4 e 3, 4 e 2, 4 e 1).
3-2-1-4-5	2 comparações (3 e 2, 3 e 1).
2-1-3-4-5	1 comparação (2 e 1).
1-2-3-4-5	$(n-1)$ iterações: vetor ordenado.

Fonte: Adaptado de Oliveira (2004).

Seja, por exemplo, C_n a quantidade de comparações quando se retira o elemento a de um vetor de n elementos. Portanto, tem-se $C_n = n-1$ comparações. Com esta ideia, pode-se escrever

$$\sum_{i=1}^{n} C_i = (n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2}, \qquad (6.1)$$

comparações no pior caso. Sua eficiência (complexidade) é, assintoticamente, representada por $O(n^2)$. No melhor caso do *Bubble Sort*, quando o vetor já se encontra ordenado, teremos apenas $(n-1)$ iterações e sua eficiência é representada por $O(n)$. No pior caso e no caso médio, temos uma complexidade representada por uma função quadrática e, no melhor caso, por uma função linear.

6.2.2.2 *Selection Sort*

O termo ***Selection Sort***, de origem inglesa, significa **Ordenação por seleção**. Utiliza o método ou paradigma da comparação por seleção ou seleção direta. Nela, a ideia é ordenar a lista selecionando, em cada iteração, os menores itens e ordenando-os da esquerda para direita, no caso de ordem crescente. Quando o menor elemento da lista é localizado na 1ª posição, ocorre nova iteração para localizar o segundo menor elemento da lista na 2ª posição e assim sucessivamente, até que a lista esteja em ordem crescente.

Algumas vantagens desse algoritmo são: fácil implementação e uso de pouca memória. Entre as desvantagens destaca-se, principalmente, seu desempenho, que é ruim em todos os casos: melhor, médio e pior, sendo indicado apenas para pequenas listas. Além disso, ele não é um algoritmo estável.

Como exemplo, consideremos o vetor $B = (5,3,1,2,4)$, com $n = 5$ elementos, o qual será ordenado de acordo com os seguintes procedimentos do *Selection Sort*.

Passo I. 5 compara com 3, 3 é menor e troca de posição com 5, assim 3 vai para a 1ª posição;
Passo II. 3 compara com 1, 1 é menor e troca de posição com 3, ou seja, 1 fica na 1ª posição e 3 na 3ª posição;
Passo III. 1 compara com 2, 1 é menor e continua sem trocar;
Passo IV. 1 compara com 4; 1 é menor e continua sem trocar.
O 1 é o menor elemento e ocupa a 1ª posição, após isso temos os seguintes passos:
Passo V. 5 compara com 3, 3 é menor e muda de posição com 5;
Passo VI. 3 compara com 2, 2 é menor e muda de posição com 3;
Passo VII. 2 compara com 4, 2 é menor e não muda de posição.
O 2 irá para a 2º posição no lugar do 3, e o 3 irá para a 4ª posição.
Seguem-se os passos:
Passo VIII. 5 compara com 3, 3 é menor e muda de posição com 5;
Passo IX. 3 compara com 4, 3 é menor e não muda de posição.
O 3 irá para a 3ª posição no lugar do 5, que está na posição inicial do 1, e o 5 vai para a 4ª posição, onde estava o 3. Realizada a troca, teremos a última comparação.
Passo X. 5 compara com 4, 4 é menor e muda de posição com 5. O 4 vai para a 4ª posição e o 5 para a 5ª posição.

A representação dos procedimentos realizados resume-se na tabela abaixo.

Utilizando o mesmo raciocínio da equação (6.1), para um vetor de n

Tabela 6.2: Exemplo do *Selection Sort* com 5 elementos, $n = 5$, representando o caso médio.

VETOR	**Nº de comparações**
5-3-1-2-4	4 comparações (5 e 3, 3 e 1, 1 e 2, 1 e 4)
1-5-3-2-4	3 comparações (5 e 3, 3 e 2, 2 e 4)
1-2-5-3-4	2 comparações (5 e 3, 3 e 4)
1-2-3-5-4	1 comparação (5 e 4)
1-2-3-4-5	$(n-1)$ iterações: vetor ordenado

Fonte: Adaptado de Oliveira (2004).

elementos, temos o total de Comparações C_n dado por:

$$\sum_{i=1}^{n} C_i = (n-1) + (n-2) + ... + 1 = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2}. \qquad (6.2)$$

Sua eficiência é representada assintoticamente por $O(n^2)$ em qualquer caso e, por isso, é considerado de péssimo desempenho em comparação com outros algoritmos.

6.2.2.3 *Quick Sort*

Será visto nesta seção um Algoritmo de Ordenação chamado ***Quick Sort***. Ele utiliza uma estratégia de ordenação por comparação bastante diferente das duas abordadas anteriormente. Ela é conhecida por divisão e conquista. Nesse tipo de Algoritmo de Ordenação, o vetor é dividido em duas partes, a partir de um elemento denominado pivô, sendo colocados, antes do pivô, os elementos menores ou iguais ao mesmo e, depois dele, os elementos maiores. O processo se repete em cada uma das partes sucessivamente até que se obtenha partes com apenas um elemento. Para finalizar, acontece o que se chama de conquista: os resultados são combinados e obtém-se a ordenação requerida. É um algoritmo bastante

popular, considerado rápido e eficiente. Utiliza técnicas de recursão sendo considerado complexo e não estável. A escolha do pivô, bem como a forma de particionamento, são situações que não serão levadas em conta neste estudo, interessando apenas a análise e comparação com os demais algoritmos acima. No esquema a seguir, apresenta-se um exemplo de como seria a ordenação utilizando a estratégia de divisão e conquista do ***Quick Sort***.

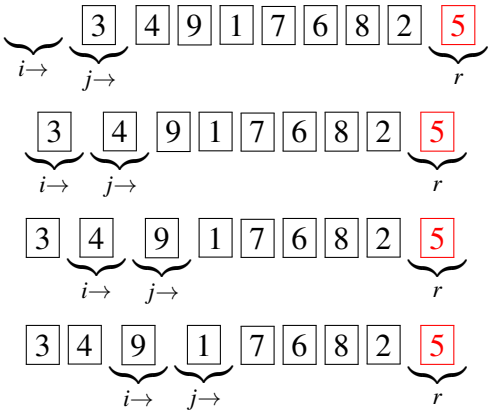
Dado o vetor $Q = (5,3,4,9,1,7,6,8,2)$, sendo $n = 9$. Uma das possíveis ordenações utilizando o ***Quick Sort*** seria:

5	3	4	9	1	7	6	8	2

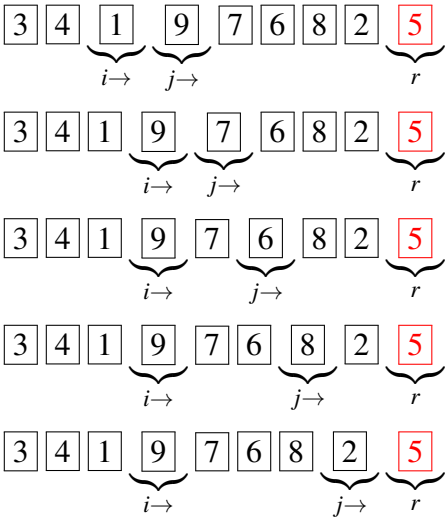
Considere 5 o pivô escolhido.

Considere r a última posição dos elementos que formam o vetor e $p \neq r$ qualquer outra posição que um elemento ocupa no vetor. Após a escolha do pivô, este fica na posição r e, para que os valores menores que o pivô fiquem antes dele e os valores maiores que ele fiquem depois, as comparações ocorrem da seguinte maneira: são criadas duas variáveis i e j representando as posições dos valores. Essas variáveis serão usadas para percorrer o vetor de entrada de modo que cada posição i seja percorrida da esquerda para direita, para localizar elementos com valores menores que o pivô, e cada posição j seja também percorrida da esquerda para a direita, até localizar elementos maiores que o pivô. Inicia-se com $i = j - 1$. Ao mesmo tempo em que j percorre o vetor, na busca dos valores maiores que o pivô, i vai, logo após, localizando os valores menores que o pivô. À medida que forem localizando, respectivamente, valores maiores e menores, eles passam para a posição imediatamente posterior até que j localize um valor menor que o pivô e i localize um valor maior que o pivô. Nesse caso, o valor de posição j troca de lugar com o valor de posição i. O procedimento continua até que todos os valores menores que o pivô estejam numa posição $p \leq i$ e os valores maiores que o pivô estejam numa posição $i < p \leq j$. Feito isso, o pivô ocupará a posição $i + 1$.

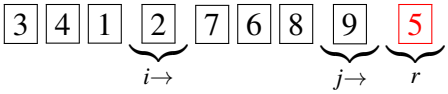
Com relação ao vetor Q teremos:



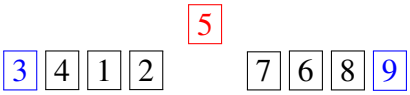
Aqui, o valor 1 muda de posição com o valor 9 e continuamos com o vetor no seguinte formato:



Observe que ocorreu, simultaneamente, de j localizar um valor menor que o pivô 5 e i localizar um valor maior que o pivô 5. Daí, o valor 2 muda de posição com o valor 9, ficando o vetor com nova configuração.

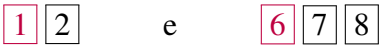


Temos que j está na posição $r-1$ e já encerrou o percurso pelo vetor, de modo que o valor na posição i também não sofrerá mais nenhuma troca com algum valor de posição j. Assim, o pivô 5 de posição r ocupará a posição $i+1$ e ficaremos com dois subvetores: o primeiro antes do pivô, com elementos menores que 5 e o segundo, depois do pivô, com elementos maiores que 5.



Escolhemos agora, nos subvetores, os pivôs 3 e 9, respectivamente, ocorrendo o mesmo procedimento nos subvetores quando o pivô foi 5. A configuração nos dois subvetores fica:

Agora escolhemos nos subvetores 1 2 e 7 6 8 os pivôs 1 e 6, respectivamente. Novamente a estratégia de divisão é efetuada e os subvetores ficam com a seguinte ordem:



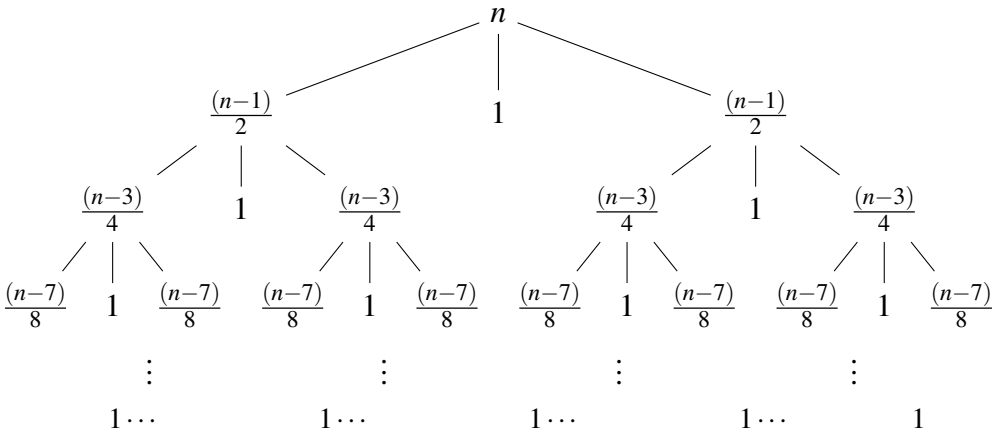
Apesar de o vetor já estar ordenado, os subvetores sofrem subdivisões até que se obtenha o último subvetor com tamanho unitário. Desse modo, no subvetor não unitário, 7 8, escolhemos o pivô 7. Executando novamente o procedimento de divisão do *Quick Sort*, fica o vetor ordenado da seguinte maneira:

Observe que a escolha do pivô é aleatória, mas o pivô ideal seria aquele que dividisse o vetor ou subvetor em tamanhos aproximadamente iguais, ou seja, um pivô que represente um valor mediano do vetor ou subvetor. Contudo, isso não ocorre na prática, e a escolha aleatória dos pivôs acarretam

nas situações de melhor caso, pior caso ou caso médio. No melhor caso, o particionamento produz segmentos(subvetores) de tamanhos iguais; já o pior caso ocorre quando o pivô é o maior (ou menor) elemento do vetor ou subvetor. O caso médio ocorre quando o particionamento é desbalanceado, como, por exemplo, numa partição em que todos os subproblemas estejam na proporção 1 para 8 em cada nível.

A seguir, faz-se uma abordagem matemática na análise da complexidade, para o melhor caso do *Quick Sort*, que se mostre clara e acessível para o Ensino Médio e auxilie os professores na obtenção da complexidade de tal algoritmo. A comparação por árvore de distribuição binária foi a escolha que se julgou mais viável para se conjecturar a função que representa o desempenho do ***Quick Sort***.

Figura 6.1: Árvore de distribuição binária do *Quick Sort*, no melhor caso, para um vetor de n elementos.



Fonte: Adaptado de Silva (2013).

Na tabela a seguir, apresenta-se uma compilação do que está representado na árvore anterior, com o intuito de facilitar a obtenção da função que representa o desempenho desse Algoritmo de Ordenação.

Por questões didáticas, não detalharemos o procedimento matemático que traduz a análise assintótica do *Quick Sort*, apenas mostraremos os

Tabela 6.3: Resumo da distribuição binária do *Quick Sort* representada na Figura (6.1).

Níveis(altura)	**Nº de subproblemas**	Tamanho do subproblema
0	1	n
1	2	$(n-1)/2$
2	4	$(n-3)/4$
3	8	$(n-7)/8$
$\vdots$	$\vdots$	$\vdots$
j	2^j	$\frac{[n-(2^j-1)]}{2^j}$

Fonte: Adaptado de Silva (2013).

resultados obtidos e deixaremos a análise dos detalhes em Santos (2020, p. 60).

Sabendo-se o valor de $\lfloor j \rfloor$, é possível calcular o trabalho total, indicado por $T(j)$, bastando somar o trabalho por nível ao longo dos níveis, isto é:

$$T(j) = \sum_{i=1}^{\lfloor j \rfloor}(n-2^i+1) = \sum_{i=1}^{\left\lfloor \log_2 \frac{(n+1)}{2} \right\rfloor}(n-2^i+1) = \underbrace{\sum_{i=1}^{\left\lfloor \log_2 \frac{(n+1)}{2} \right\rfloor} n}_{I} - \underbrace{\sum_{i=1}^{\left\lfloor \log_2 \frac{(n+1)}{2} \right\rfloor} 2^i}_{II} + \underbrace{\sum_{i=1}^{\left\lfloor \log_2 \frac{(n+1)}{2} \right\rfloor} 1}_{III}. \tag{6.3}$$

Em cada parcela anterior, a que possui maior ordem de crescimento é (I), ficando a complexidade do trabalho total $T(j)$ realizado pelo Algoritmo *Quick Sort*, para o melhor caso, representada assintoticamente por $O(n \cdot \log_2 n)$ ou $O(n \cdot \log n)$.

6.2.2.4 Comparação gráfica das complexidades dos algoritmos de ordenação

Observou-se, nas seções antecedentes, algumas leis algébricas de funções representando o total de comparações, bem como as complexidades dos Algoritmos de Ordenação analisados. Faremos uma comparação de tais desempenhos inicialmente numa tabela e posteriormente por gráfico. A visualização gráfica expressa bem o comportamento assintótico das funções que os representam. Vale salientar que a utilização de linguagens variadas, para expressar os resultados matemáticos obtidos ao se analisar os algoritmos, é bastante relevante para que se desenvolvam as competências requeridas na BNCC.

Tabela 6.4: Tabela de comparação do desempenho assintótico dos Algoritmos de Ordenação

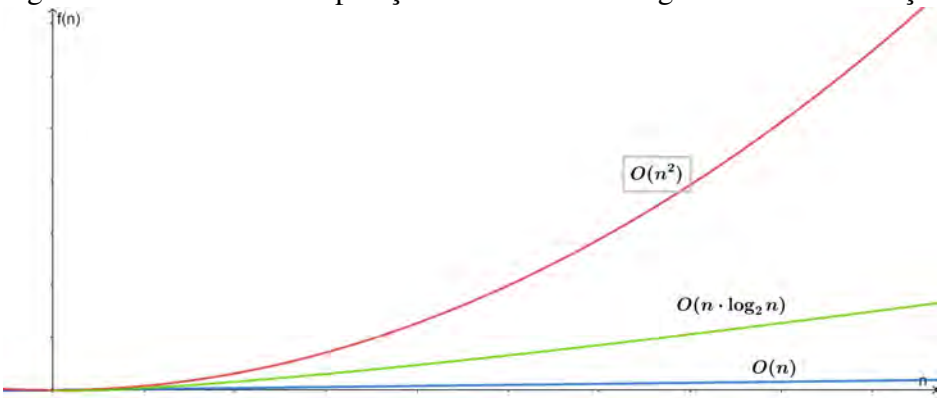
Nº de elementos	***Bubble Sort*:** $O(n)$	***Selection Sort*:** $O(n^2)$	***Quick Sort*:** $O(n \cdot \log_2 n)$
2	2	4	2
4	4	16	8
16	16	256	64
256	256	65.536	2048
1.024	1.024	1.048.576	10.240

Fonte: Elaborada pelo autor.

É notório na tabela anterior que, no melhor caso, o *Bubble Sort* é sempre mais eficiente que os outros dois Algoritmos de Ordenação, mas o *Quick Sort* se mostra mais eficiente que um algoritmo de complexidade Quadrática.

A complexidade $O(n \cdot \log n)$, chamada de linearítmica, possui uma taxa de crescimento maior que uma complexidade linear $O(n)$ e menor que qualquer complexidade polinomial, como é o caso de $O(n^2)$. Sabe-se que a análise assintótica *Big-O* compara as funções no limite superior, ou seja, quando a quantidade n de elementos do vetor é suficientemente grande.

Figura 6.2: Gráfico de comparação assintótica dos Algoritmos de Ordenação



Fonte: Elaborada pelo autor.

Um algoritmo é dito mais eficiente que outro quando sua complexidade possui assintoticamente uma taxa de crescimento menor, portanto, conclui-se que $O(n) < O(n \cdot \log n) < O(n^2)$.
O caso médio do *Quick Sort* se aproxima do melhor caso, tendo sua complexidade $O(n \cdot \log n)$ também e, apesar do seu desempenho no pior caso ser $O(n^2)$, na média, sua performance é excelente, o que o faz ser tão popular.

6.2.3 Sequência didática

O objetivo da sequência didática foi propor as atividades de modo a abordar os conteúdos matemáticos concernentes aos Algoritmos de Ordenação e desenvolver, gradualmente, os conceitos ligados ao tema, bem como as competências e habilidades em matemáticas da BNCC.

No capítulo 4 de Santos (2020), propomos uma sequência didática com oito aulas, mas fizemos um recorte apenas para ilustrar a forma como se propõe a abordagem do tema no Ensino Médio. Escolhemos a aula 5 a seguir:

5ª aula: Ordenando com os Algoritmos *Bubble Sort*, *Selection Sort*

e *Quick Sort*. *Procedimentos A turma será dividida em equipes que receberão a Ficha 3 (ver modelo sugerido) e cinco fichas ou cartas de baralho numeradas. As fichas deverão ser ordenadas de três formas diferentes, utilizando as estratégias do *Bubble Sort*, *Selection Sort* e *Quick Sort* respectivamente, registrando todo o procedimento. O professor pede para cada equipe socializar suas respostas e comparar com as estratégias que foram criadas pelos estudantes em outra aula. Os estudantes ainda não estarão instigados a desenvolver a modelagem e escrever algebricamente a função que representa a complexidade. Se limitarão a executar o procedimento de cada algoritmo.

Ficha 3

1) Cada equipe está recebendo cinco cartas com valores diferentes. Como por exemplo: 4 -2 0 8 5. Estas cartas deverão ser ordenadas utilizando os Algoritmos de Ordenação a seguir e registrar todo o processo em um material a parte.

Ordenando com o *Bubble Sort*: Se o objetivo é ordenar os valores em forma crescente, então, a posição atual é comparada com a próxima posição e, se a posição atual for maior que a posição posterior, é realizada a troca dos valores nessa posição. Caso contrário, não é realizada a troca, apenas passa-se para o próximo par de comparações.

Ordenando com o *Selection Sort*: A ordenação por seleção ou *Selection Sort* consiste em selecionar, por comparação, o menor item e colocar na primeira posição, selecionar o segundo menor item e colocar na segunda posição, segue estes passos até que reste um único elemento.

Ordenando com o *Quick Sort*: O vetor é dividido em duas partes a partir de um elemento denominado pivô, antes do pivô são colocados os elementos menores ou iguais ao mesmo e depois dele os elementos maiores. O processo se repete em cada uma das partes (subvetores) sucessivamente até que se obtenha partes com apenas um elemento. Para finalizar, os resultados são combinados e obtém-se o resultado esperado.

As competências e habilidades relacionadas a essa aula (atividade) são:

6.2.3.1 Competência específica 3 da BNCC

Utilizar estratégias, conceitos, definições e procedimentos matemáticos para interpretar, construir modelos e resolver problemas em diversos contextos, analisando a plausibilidade dos resultados e a adequação das soluções propostas, de modo a construir argumentos consistentes.

6.2.3.2 Habilidade 15

Investigar e registrar, por meio de um fluxograma, quando possível, um algoritmo que resolve um problema.

6.2.3.3 Competência específica 4 da BNCC

Compreender e utilizar com flexibilidade e precisão diferentes registros de representação matemáticos (algébrico, geométrico, estatístico, computacional, etc.) na busca de solução e comunicação de resultados de problemas.

6.2.3.4 Habilidade 05

Utilizar conceitos iniciais de uma linguagem de programação na implementação de algoritmos escritos em linguagem corrente e/ou matemática. Além desse tipo de atividade apresentado na 5ª aula da referida sequência, outras podem ser propostas com o tema envolvendo jogos e brincadeiras, modelagem matemática e representação gráfica. Como foi apresentado na Seção 6.2, o tema Algoritmos de Ordenação propicia o desenvolvimento de competências e habilidades relacionadas a diversos conteúdos matemáticos.

6.3 Considerações finais

O estudo em tela mostrou que é possível uma abordagem didática dos Algoritmos de Ordenação no Ensino Médio sem a presença de estruturas de programação computacional, e que essa abordagem favorece o desenvolvimento de competências e habilidades que constam na BNCC. São vários os conteúdos matemáticos relacionados com o tema como Números, Álgebra e Geometria. Apesar de termos suprimidos o conteúdo matemático mais denso nesse artigo, o tema engloba o estudo e aprofundamento de séries aritméticas e geométricas, função afim, função quadrática e função logarítmica.

A notação assintótica **O - grande** não se enquadra no Ensino Médio, mas pode ser apresentada complementarmente na abordagem do tema como um ganho na aprendizagem. O tema aqui proposto é recorrente nos cursos de graduação em computação, mas como consta na BNCC, que é um documento recente, sua inclusão no Ensino Médio se faz necessária e se mostra viável. Daí, é importante que o tema possa ser mais pesquisado e

aprofundado pelo meio acadêmico e que mais subsídios como sequências didáticas possam não apenas serem construídas, mas também possam ser aplicadas e seus impactos educacionais avaliados.

6.4 Referências bibliográficas

BRASIL. **Parâmetros Curriculares Nacionais de Matemática**. Brasília: MEC, 1998. 148 p.

BRASIL. **Parâmetros Curriculares Nacionais do Ensino Médio (PCNEM).** Brasília: MEC, 2000. 148 p.

BRASIL. **PCN+ Ensino médio: Orientações educacionais complementares aos Parâmetros Curriculares Nacionais-Ciências da Natureza, Matemática e suas Tecnologias.** Brasília: MEC, 2002. 142 p.

BRASIL, M. d. E. **Base Nacional Curricular Comum.** Brasil: MEC, 2018.

CORMEN, T. H. et al. **Algoritmos: Teoria e Prática, tradução da segunda edição, [americana].** São Paulo: Editora Campus, 2002.

OLIVEIRA, P. R. de. **Algoritmos e Programação de Computadores.** 2004. Disponível em: <https://slideplayer.com.br/slide/364011/>.

PERNAMBUCO, C. **Parâmetros para a Educação Básica do Estado de Pernambuco.** Recife-PE: Governo de Pernambuco, 2012.

RIBEIRO, M. R. da C. **Grafos, Algoritmos e Programação.** 152 p. Dissertação (Mestrado) — UFRPE, Recife-PE, 2018.

SANTOS, I. F. dos. **Algoritmos de Ordenação: uma abordagem didática para o ensino médio.** 78 p. Dissertação (Mestrado) — UFRPE, Recife-PE, 2020.

SILVA, A. P. C. da. **Classificação de dados por troca QuickSort.** 2013. Disponível em: <https://slideplayer.com.br/slide/364011/>.