**Nome: Dhruv Babani (22110019-1)**

**Algoritmos e Estruturas de Dados I – T.10**

**Trabalho 1 – Análise de Algoritmos**

# Introdução

Neste trabalho, analisou-se o comportamento de crescimento de 5 algoritmos, a fim de definir, aproximadamente, o custo operacional de cada algoritmo e o tipo de função com a qual seu comportamento assemelha-se. Para adquirir os dados para a análise, executou-se os algoritmos em VSCode, com valor inicial igual a 0 e final igual a 100 com incremento de 5 unidades por repetição(exceção do Algoritmo 4,com apenas até o n = 55), e construiu-se suas respectivas tabelas e gráficos em Excel.

# A análise dos algoritmos

 **Algoritmo 1:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n f(n) | |  | n f(n) | |
| 0 | 1 | 65 | 1271974 |
| 5 | 164 |  | 70 | 1695174 |
| 10 | 1369 |  | 75 | 2216065 |
| 15 | 5420 |  | 80 | 2848721 |
| 20 | 15081 |  | 85 | 3608044 |
| 25 | 34034 |  | 90 | 4509909 |
| 30 | 66904 |  | 95 | 5571080 |
| 35 | 119235 |  | 100 | 6809401 |
| 40 | 197561 |  |
| 45 | 309304 |  |
| 50 | 462839 |  |
| 55 | 667450 |  |
| 60 | 933441 |  |

Após a execução do algoritmo e a construção da tabela com seus dados de execução, construiu-se o gráfico a seguir:

Mesmo com a formação de um gráfico , o que indica a possibilidade de o comportar-se como uma função cúbica, realizaram-se testes para definir seu tipo de função característica com maior precisão

Primeiramente, foi feita a transformação do eixo “*f(n)*” para a escala logarítmica, e então construiu--se o seguinte gráfico de eixos (*n*, *log(f(n)*) para determinar se o Algoritmo 1 comporta-se, ou não, como uma função exponencial (*a \* bn*):

Como o gráfico formado não representa uma reta, confirmou-se que o Algoritmo 1 consome operações seguindo uma função polinomial. Então, foram escolhidos 2 valores obtidos com o algoritmo para calcular o valor do expoente “*b*” de sua função:

𝑓1(10) = 1369

𝑓1(100) = 6809401

Para o cálculo, foram usados os logaritmos dos valores de “*f1*” e dos valores de “*n*” associados a eles, obteve-se:

B = log 6809401 – log 1369/ log 100 – log 10 = 3.69

Portanto, sugere-se que a função do Algoritmo 1 cresce como uma função polinomial do grau 3: 𝑓1(𝑛) ≈4

 **Algoritmo 2:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| n f(n) | |  | n f(n) | |
| 0 | 1 | 65 | 64 |
| 5 | 4 |  | 70 | 69 |
| 10 | 9 |  | 75 | 74 |
| 15 | 14 |  | 80 | 79 |
| 20 | 19 |  | 85 | 84 |
| 25 | 24 |  | 90 | 89 |
| 30 | 29 |  | 95 | 94 |
| 35 | 34 |  | 100 | 99 |
| 40 | 39 |  |
| 45 | 44 |  |
| 50 | 49 |  |
| 55 | 54 |  |
| 60 | 59 |  |

Após a execução do algoritmo e a construção da tabela com seus dados de execução, construiu-se o gráfico a seguir:

Em razão do alto crescimento do gráfico e de sua representação em reta, realizou-se a análise de seus dados para determinar se o Algoritmo 2 comporta-se como uma função exponencial (*a \* bn*). Então, converteu-se o eixo “*f(n)*” para a escala logarítmica e construiu-se o seguinte gráfico:

Como o gráfico formado representa uma reta, nota-se que o Algoritmo 2 consome operações seguindo uma função polinomial. Então, foram escolhidos 2 valores obtidos com o algoritmo para determinar o valor do expoente “*b*” de sua função:

𝑓2(10) = 9

𝑓2(100) = 99

Tomando logaritmos dos valores de “*f2*” e dos valores de “*n*” associados a eles, obteve-se:

B = log 99 – log 9/log 100 - log 10 = 1.04

Portanto, sugere-se que a função do Algoritmo 2 cresce como uma função polinomial do 1 grau: 𝑓2(𝑛) ≈ 1

 **Algoritmo 3:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n f(n) | | |  | n f(n) | |
| 0 | | 0 | 65 | 6753 |
| 5 | | 52 |  | 70 | 7800 |
| 10 | | 191 |  | 75 | 8923 |
| 15 | | 412 |  | 80 | 10116 |
| 20 | | 708 |  | 85 | 11392 |
| 25 | | 1078 |  | 90 | 12744 |
| 30 | | 1525 |  | 95 | 14165 |
| 35 | | 2047 |  | 100 | 15662 |
| 40 | | 2644 |  |
| 45 | | 3318 |  |
| 50 | | 4060 |  |
| 55 | | 4883 |  |
| 60 | | 5782 |  |
|  |

Após a execução do algoritmo e a construção da tabela com seus dados de execução, construiu o gráfico a seguir:

Devido ao crescimento do gráfico ter representação em forma de curva, converteu-se seu eixo “*f(n)*” para a escala logarítmica para analisar se o Algoritmo 3 comporta-se como uma função exponencial (*a \* bn*). Então, e construiu-se o seguinte gráfico:

Como o gráfico formado representa uma reta, confirmou-se que o Algoritmo 3 consome operações seguindo uma função polinomial. Então, foram escolhidos 2 valores obtidos com o algoritmo para calcular o valor do expoente “*b*” de sua função:

𝑓3(10) = 191

𝑓3(100) = 15662

Para o cálculo, foram usados os logaritmos dos valores de “*f3*” e dos valores de “*n*” associados a eles, obteve-se:

B = log 15662 – log 192 /log 100 – log 10 = 1.91

Portanto, sugere-se que a função do Algoritmo 3 cresce como uma função polinomial do 2º grau: 𝑓3(𝑛) ≈ 2

 **Algoritmo 4:**

Após a execução do algoritmo e a construção da tabela com seus dados de execução, construiu o gráfico a seguir:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | n f(n) | | |  |  | | 0 | | 0 | | 5 | | 935 |  | | 10 | | 57960 |  | | 15 | | 678930 |  | | 20 | | 3828595 |  | | 25 | | 14764050 |  | | 30 | | 44190780 |  | | 35 | | 112059920 |  | | 40 | | 250068390 |  | | 45 | | 508695165 |  | | 50 | | 958180300 |  | | 55 | | 1701373410 |  | |  | |  |

Em razão do altíssimo crescimento do gráfico e de sua representação em curva, realizou-se a análise de seus dados para determinar se o Algoritmo 4 comporta-se como uma função exponencial (*a \* bn*). Então, converteu-se o eixo “*f(n)*” para a escala logarítmica e construiu-se o seguinte gráfico:

Como o gráfico formado representa uma reta, confirmou-se que o Algoritmo 4 consome operações seguindo uma função polinomial. Então, foram escolhidos 2 valores obtidos com o algoritmo para calcular o valor do expoente “*b*” de sua função:

𝑓4(10) = 57960

𝑓4(50) = 958180300

Para o cálculo, foram usados os logaritmos dos valores de “*f4*” e dos valores de “*n*” associados a eles, obteve-se:

B= log 958180300 – log 57960 /log 50 – log 10 = 6.0350

Portanto, sugere-se que a função do Algoritmo 4 cresce como uma função polinomial do 6º grau: 𝑓4(𝑛) ≈ 6

 **Algoritmo 5:**

Após a execução do algoritmo e a construção da tabela com seus dados de execução, construiu o gráfico a seguir:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | n f(n) | |  | n f(n) | | | 0 | 0 | 65 | 4468657 | | 5 | 187 |  | 70 | 6009456 | | 10 | 2631 |  | 75 | 7918158 | | 15 | 12963 |  | 80 | 10249100 | | 20 | 40550 |  | 85 | 13060447 | | 25 | 98527 |  | 90 | 16414031 | | 30 | 203756 |  | 95 | 20375523 | | 35 | 376878 |  | 100 | 25014250 | | 40 | 642250 |  | | 45 | 1028017 |  | | 50 | 1566031 |  | | 55 | 2291943 |  | | 60 | 32545100 |  | |  |

Devido ao altíssimo crescimento do gráfico e de sua representação em forma de curva, converte-se eixo “*f(n)*” para a escala logarítmica para analisar se o Algoritmo 5 comporta-se como uma função exponencial (*a \* bn*). Então, e construiu-se o seguinte gráfico:

Como o gráfico formado representa uma reta, concluiu-se que o Algoritmo 5 consome operações seguindo uma função polinomial. Então, foram escolhidos 2 valores obtidos com o algoritmo para calcular o valor do expoente “*b*” de sua função:

𝑓5(10) = *2631*

𝑓5(100) = 25014250

Para o cálculo, foram usados os logaritmos dos valores de “*f4*” e dos valores de “*n*” associados a eles, obteve-se:

B = log 25014250 – log 2631/log 100 – log 10 = 3,97

Portanto, por aproximação, sugere-se que a função do Algoritmo 5 cresce como uma função polinomial do 2º grau:

𝑓5(𝑛) ≈ 4

# Conclusão

Algoritmos podem demandar uma grande capacidade de processamento de um computador para executar tarefas, mesmo que seus valores de entrada não sejam tão altos, e seus requerimentos podem tornar-se maiores ou menores mesmo com pequenas mudanças no desenvolvimento do código do algoritmo. Assim, a análise de algoritmos e a busca por sua otimização mostram-se de grande importância na construção de um código com execução rápida e de menor custo.

# Implementação

* Algoritmo 1
* public static int f1(int n){
* int i, j, k, res = 0;
* int cont\_op = 0;
* for( i = 1; i <= n+1; i += 1 ){
* for( j = 1; j <= i\*i; j += i+1 ){
* for( k = i/2; k <= n+j; k += 2 ) {
* res = res + n-1;
* cont\_op++;
* }
* }
* }
* return cont\_op;
* }
* Algoritmo 2
* public static int f2( int n ) {
* int i, j, k, res = 0;
* int cont\_op = 0;
* for( i = n; i <= n; i += i/2+1 ){
* for( j = i/2; j <= i\*i; j += i+1 ){
* for( k = n; k <= 2\*n; k += i+1 ) {
* res = res + n;
* cont\_op++;
* }
* }
* }
* return cont\_op;
* }
* Algoritmo 3
* public static int f3( int n ) {
* int i, j, k, res = 0;
* int cont\_op = 0;
* for(i = 1;i <= n\*n; i+= 2){
* for(j= i/2;j <= 2\*i; j+= i/2+1){
* for(k = j+1;k<= n+j;k += k/2+1){
* res = res + Math.abs(j-i);
* cont\_op++;
* }
* }
* }
* return cont\_op;
* }
* Algoritmo 4
* public static int f4( int n ) {
* int i, j, k, res = 0;
* int cont\_op = 0;
* for( i = n; i <= n\*n; i += 2 ){
* for( j = n+1; j <= n\*n; j += 2 ){
* for( k = j; k <= 2\*j; k += 2 ) {
* res = res + 1;
* cont\_op++;
* }
* }
* }
* return cont\_op;
* }
* Algoritmo 5
* public static int f5( int n ) {
* int i, j, k, res = 0;
* int cont\_op = 0;
* for( i = 1; i <= n\*n; i += 1 ){
* for( j = 1; j <= i; j += 2 ){
* for( k = n+1; k <= 2\*i; k += i\*j ) {
* res = res + k+1;
* cont\_op++;
* }
* }
* }
* return cont\_op;
* }