

Trabalho Prático - Arquitetura e Organização de computadores

Quebrando senha HPC

Dupla: Gabrielle de Oliveira Fonseca (0072379) e Maria Eduarda Rodrigues Alves Moraes (0072382)

1.Introdução

Ao receber um arquivo compactado e encriptado com uma senha de cinco caracteres, incluindo caracteres especiais e alfanuméricos, nossa tarefa foi encontrar a senha correta para descriptar o arquivo utilizando uma estratégia de força bruta baseada nos caracteres presentes na tabela ASCII. Um ponto importante é que nossa resolução deveria utilizar múltiplos núcleos de processamento paralelos, ou seja, particionar o problema e utilizar o paradigma de computação paralela para reduzir o tempo de execução da solução.

1.1.Computação paralela

A computação paralela nada mais é que o uso simultâneo de vários recursos computacionais a fim de reduzir o tempo necessário para resolver um determinado problema. Dentre os recursos computacionais, temos:

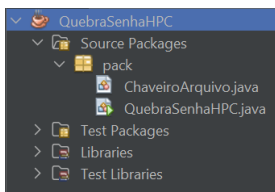
- > Um computador e múltiplos processadores;
- > Um número X de computadores ligados por uma rede;
- > Os dois recursos juntos.

Dentre as diversas vantagens da programação paralela, as que nos levaram a esse trabalho foram:

- > Reduzir o tempo necessário para solucionar um problema;
- > Resolver problemas mais complexos e de maior dimensão;
- > Ultrapassar os limites físicos de velocidade da programação sequencial.

2.Implementação

2.1.Classes do projeto



2.1.1.ChaveiroArquivo

```
15 public class ChaveiroArquivo {
16
17     private final File arquivoSegredo;
18     private String caminhoArquivo;
19
20     public ChaveiroArquivo(File arquivoSegredo) {
21         this.arquivoSegredo = arquivoSegredo;
22
23         if (this.arquivoSegredo != null) {
24             //caminho onde vamos ext. o arquivo compactado
25             this.caminhoArquivo = this.arquivoSegredo.getAbsolutePath().getParent();
26         }
27     }
28
29     //método que permite realizar um teste com a senha passada como parâmetro para um dado arquivo
30     public boolean tentaSenha(String senha) {
31         try {
32             ZipFile zipFile = new ZipFile(this.arquivoSegredo);
33
34             //esta com senha
35             if (zipFile.isEncrypted()) {
36                 //tentativa da senha aqui
37                 zipFile.setPassword(senha.toCharArray());
38             }
39             List fileHeaderList = zipFile.getFileHeaders();
40
41             //solução genérica para qualquer tamanho de Header
42             for (int i = 0; i < fileHeaderList.size(); i++) {
43                 FileHeader fileHeader = (FileHeader) fileHeaderList.get(i);
44                 //onde o arquivo será armazenado
45                 zipFile.extractFile(fileHeader, this.caminhoArquivo);
46                 //System.out.println("encontramos a senha e o arquivo");
47             }
48             return true;
49         } catch (ZipException e) {
50             //System.out.println("Erro tente novamente");
51             return false;
52         }
53     }
54 }
55 }
```

2.1.2.Main

Nós implementamos a nossa solução na main do projeto, chamada “QuebrqSenhaHPC”, tomamos o cuidado de comentar de forma geral cada um dos passos que seguimos, segue passo a passo:

1. Criamos um objeto “ChaveiroArquivo” usando um arquivo específico.
2. Calculamos o número total de senhas, que é igual a 95 elevado à quinta potência (95^5), e armazenamos na variável “numPasswords”.
3. Calculamos o tamanho de cada parte das senhas, dividindo o número total de senhas pelo número de processadores disponíveis.
4. Criamos um array de threads com o tamanho igual ao número de processadores disponíveis.
5. Inicializamos cada thread com uma parte diferente das senhas, usando o índice “i” para determinar o início e o fim de cada parte.
6. Iniciamos cada thread usando o método start().
7. Aguardamos todas as threads terminarem usando um loop “for” que usa o método “join()” em cada thread.

```
7 public class QuebraSenhaHPC extends Thread {
8
9     public static void main(String[] args) {
10         JFileChooser janela = new JFileChooser();
11
12         //monitora a ação do usuário na árvore de diretório do sistema
13         int operacao = janela.showOpenDialog(null);
14         if (operacao == JFileChooser.APPROVE_OPTION) {
15
16             //ref. do arquivo selecionado pelo user...
17             File arquivo = janela.getSelectedFile();
18
19             //será que o arquivo é ext. .zip???
20             if (!arquivo.getAbsolutePath().contains(".zip")) {
21                 JOptionPane.showMessageDialog(null, "O arquivo selecionado deve ter ext. do tipo .zip", "Arquivo incorreto", JOptionPane.WARNING_MESSAGE);
22                 System.exit(0);
23             }
24
25             ChaveiroArquivo chaveiro = new ChaveiroArquivo(arquivo);
26
27             // Divide o conjunto de senhas em quatro partes iguais
28             int numPasswords = (int) Math.pow(95, 5);
29             int chunkSize = numPasswords / Runtime.getRuntime().availableProcessors();
30
31             // Cria um array de threads
32             Thread[] threads = new Thread[Runtime.getRuntime().availableProcessors()];
33
34             // Inicializa cada thread com uma parte diferente das senhas
35             for (int i = 0; i < Runtime.getRuntime().availableProcessors(); i++) {
36                 final int start = i * chunkSize;
37                 final int end = (i == Runtime.getRuntime().availableProcessors() - 1) ? numPasswords : (i + 1) * chunkSize;
38                 threads[i] = new QuebraSenhaThread(start, end, chaveiro);
39                 threads[i].start();
40             }
41
42             // Aguarda todas as threads terminarem
43             for (Thread thread : threads) {
44                 try {
45                     thread.join();
46                 } catch (InterruptedException e) {
47                     // Ignora interrupções
48                 }
49             }
50
51             // "Chaveiro(s)" começa a trabalhar aqui ;)
52             ChaveiroArquivo chaveiro = new ChaveiroArquivo(arquivo);
53
54             /* //executa se a senha estiver correta (a senha tem tamanho de 5 caracteres)
55             if (trab.tentaSenha("Chute de uma senha, você pode alterar isso aqui")) {
56                 System.out.println("Senha correta");
57             } */
58         } else {
59             JOptionPane.showMessageDialog(null, "O arquivo não foi selecionado", "Arquivo???", JOptionPane.WARNING_MESSAGE);
60         }
61     }
62 }
63
64 public static volatile boolean senhaEncontrada = false; //criando um boolean para que assim que uma das threads encontrar a senha, todas parem
65
66 @Override
67 public void run() {
68     for (int i = start; i < end; i++) {
69         String password = toPassword(i);
70         if (chaveiro.tentaSenha(password)) {
71             System.out.println("Senha correta: " + password);
72             senhaEncontrada = true;
73             stopAllThreads(Thread.currentThread().getThreadGroup());
74             break;
75         }
76     }
77     if (senhaEncontrada) {
78         System.exit(0);
79     }
80 }
81
82 private static String toPassword(int index) {
83     char[] password = new char[5];
84     for (int i = 0; i < 5; i++) {
85         password[i] = (char) ((index % 94) + 33); // O intervalo é de 33 a 126, então são 94 caracteres possíveis
86         index /= 94;
87     }
88     return new String(password);
89 }
90
91
92
93
94
95
96
97
98
99
100
101 }
```

```

103 // Interrompe todas as threads
104 private void stopAllThreads(ThreadGroup threadGroup) {
105     // Obtém a quantidade de threads ativas no grupo
106     int activeThreadCount = threadGroup.activeCount();
107     // Cria um array para armazenar as threads ativas
108     Thread[] threads = new Thread[activeThreadCount];
109     // Preenche o array com as threads ativas
110     threadGroup.enumerate(threads);
111     // Itera sobre o array de threads, interrompendo cada uma delas
112     for (Thread thread : threads) {
113         thread.interrupt();
114     }
115 }
116 }
117

```

3. Testes

Para otimizar o tempo de implementação e tentar alcançar o resultado mais rápido, tendo em vista o objetivo principal do projeto, quebrar a senha do "arquivo.zip" enviado pelo professor, executamos diversos testes com senhas menos complexas. Desta forma, foi criado um único arquivo chamado "Teste.zip", para que pudéssemos testar diversos níveis de senhas. Nesse caso, utilizamos apenas o caractere "!", encontrado na posição 33 da Tabela ASCII, posição inicial estabelecida pelo professor.

Tabela ASCII					
= 32	! = 33	" = 34	# = 35	\$ = 36	
% = 37	& = 38	' = 39	(= 40) = 41	
* = 42	+ = 43	, = 44	- = 45	. = 46	
/ = 47	0 = 48	1 = 49	2 = 50	3 = 51	
4 = 52	5 = 53	6 = 54	7 = 55	8 = 56	
9 = 57	: = 58	; = 59	< = 60	= = 61	
> = 62	? = 63	@ = 64	A = 65	B = 66	
C = 67	D = 68	E = 69	F = 70	G = 71	
H = 72	I = 73	J = 74	K = 75	L = 76	
M = 77	N = 78	O = 79	P = 80	Q = 81	
R = 82	S = 83	T = 84	U = 85	V = 86	
W = 87	X = 88	Y = 89	Z = 90	[= 91	
\ = 92] = 93	^ = 94	_ = 95	` = 96	
a = 97	b = 98	c = 99	d = 100	e = 101	
f = 102	g = 103	h = 104	i = 105	j = 106	
k = 107	l = 108	m = 109	n = 110	o = 111	
p = 112	q = 113	r = 114	s = 115	t = 116	
u = 117	v = 118	w = 119	x = 120	y = 121	
z = 122	{ = 123	= 124	} = 125	~ = 126	

3.1. Senha "!!!"

```

Output - QuebraSenhaHPC (run)
run:
Senha correta: !!
BUILD SUCCESSFUL (total time: 7 seconds)

```

3.2. Senha "!!!!"

```

Output - QuebraSenhaHPC (run)
run:
Senha correta: !!!!
BUILD SUCCESSFUL (total time: 6 seconds)

```

3.3. Senha "!!!!!"

```

Output - QuebraSenhaHPC (run)
run:
Senha correta: !!!!!
BUILD SUCCESSFUL (total time: 6 seconds)

```

3.4. Senha "!!!!!!"

```

Output - QuebraSenhaHPC (run)
run:
Senha correta: !!!!!!
BUILD SUCCESSFUL (total time: 12 seconds)

```

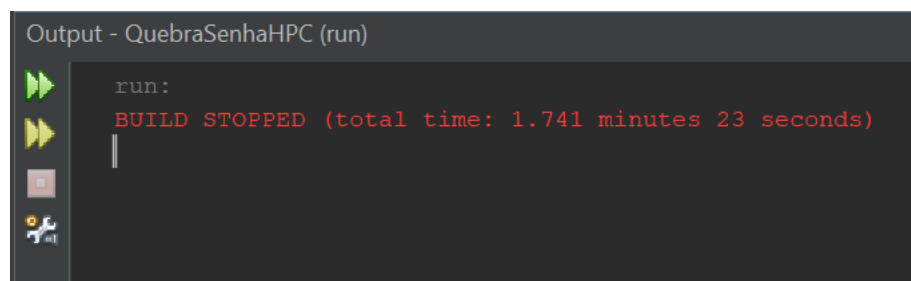
3.5. Conclusão dos testes

Após executar esses testes diversas vezes, pudemos modificar o código de forma a otimizá-lo o máximo que conseguimos, chegando a esses resultados apresentados acima. Com base nesses testes, concluímos que o código conseguiria quebrar a senha do "arquivo.zip", porém demandaria muito mais tempo do que o esperado. Nos primeiros testes, antes da otimização, o programa foi interrompido após 38 horas, rodando sem interrupções.

4. Resultados

Com mais de 15 dias de trabalho, diversos testes, conquistas e frustrações, não conseguimos chegar a um resultado final muito satisfatório, porém, com os testes apresentados anteriormente, podemos chegar a algumas conclusões.

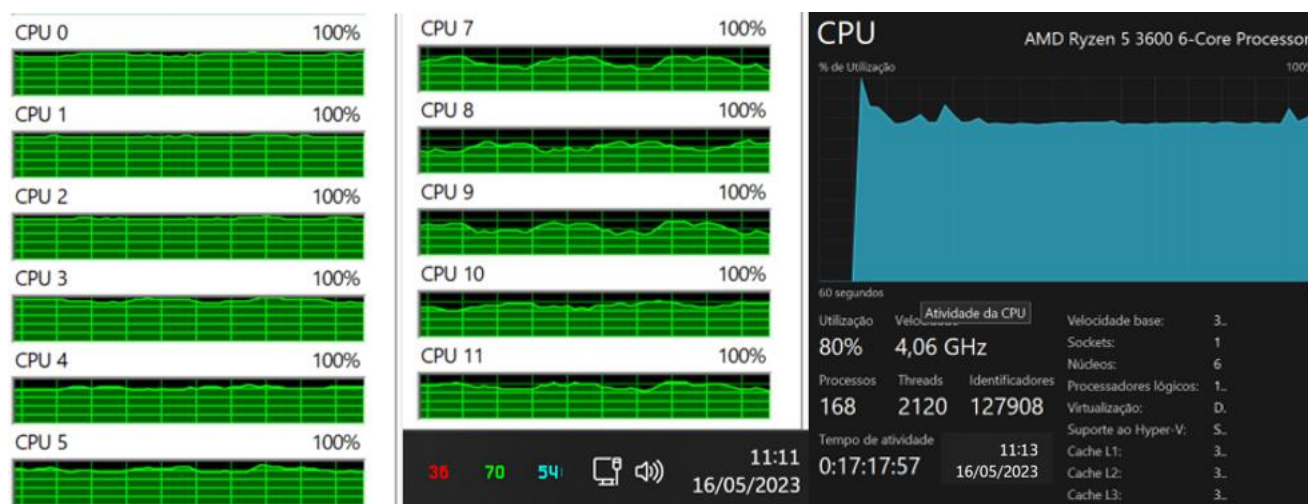
Nosso último teste foi executado no dia 15/05/2023 às 18:00 e foi interrompido no dia 16/05/2023 às 23:00, durando exatamente 29 horas, mas não obtivemos sucesso pela falta de tempo para a entrega.



```
Output - QuebraSenhaHPC (run)

run:
BUILD STOPPED (total time: 1.741 minutes 23 seconds)
```

O uso da CPU total chegou a quase 80% durante todo o processo, além de estar usando todas as CPUs, como solicitado.



5. Conclusão

Nosso código não chegou ao resultado final de extrair o "arquivo.zip" com a senha secreta, mesmo que os testes tenham mostrado que seria possível, talvez com mais alguns dias rodando. Apesar de não alcançarmos o objetivo com sucesso, observamos algumas modificações que poderiam ser feitas no projeto, a fim de otimizá-lo ainda mais e conseguir quebrar essa senha em menos tempo do que o executado, para isso seriam necessários mais testes e mais dias para a entrega.

Esse com certeza será um projeto com o qual vamos mexer por mais tempo, com o passar dos períodos, para que no final, tenhamos um código completo e otimizado, com todos os conceitos apresentados no curso até aquele momento.

Encontramos grande grau de dificuldade na solução deste exercício, mas foi um ótimo trabalho para nos desafiar e nos ensinar na prática como solucionar problemas com a computação paralela. Esperamos que, mais para a frente, possamos alcançar o resultado esperado com esse projeto, concluindo com todos os resultados esperados.

6. Bibliografia

Ricardo Rocha DCC-FCUP Programação Paralela e Distribuída 2007/08 Fundamentos 1 Programação Paralela e Distribuída Fundamentos Porquê Programação Paralela? Porquê Programação Paralela? . [s.l: s.n.]. Disponível em: <<https://www.dcc.fc.up.pt/~ricroc/aulas/0708/ppd/apontamentos/fundamentos.pdf>>. Acesso em: 16 maio. 2023.

LINK, G. et al. Gerando tabela ASCII de base decimal. Disponível em: <<https://www.shellscriptx.com/2016/11/gerando-tabela-ascii-de-base-decimal.html>>. Acesso em: 16 maio. 2023.