Trabalho apresentado no XL CNMAC, Evento Virtual - Co-organizado pela Universidade do Mato Grosso do Sul (UFMS).

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Estudo quantitativo de algoritmos para teste de primalidade

David Vinicius da Silva[1]
IFSP/Câmpus Hortolândia
Paulo Eduardo Nogueira[2]
IFSP/Câmpus Hortolândia

Por conta de sua natureza, números primos são utilizados para criptografar informações sensíveis que não podem ser compartilhadas com terceiros, como senhas ou tokens. Sendo assim, pode se entender a importância de se descobrir novos números primos, pois, quanto maior o número e a sua exclusividade, maior a integridade da chave, garantindo mais segurança na codificação dos dados [1].

Contudo, determinar se um número é primo, dependendo da situação, pode não ser uma tarefa trivial. Há, portanto, algumas formas de se atestar se um número é primo ou não, a mais óbvia é pela definição, ou seja, a contagem da quantidade de divisores de um número.

No entanto, essa abordagem não é a melhor estratégia quando se trata de números muito grandes, por exemplo, o último maior número primo encontrado tem cerca de quase 25 milhões de dígitos [4]. Contar o número de divisores de algo dessa magnitude pode levar a uma grande quantia de tempo. Para estes casos, estratégias mais eficazes, como algoritmos, torna-se necessárias para novas descobertas.

Naturalmente, haverá algoritmos que são mais rápido do que outros e conhecer essa diferença pode ser relevante para usar recursos computacionais de maneira eficiente. Portanto, neste trabalho, objetivou-se comparar, em termos de desempenho, dois algoritmos que podem ser utilizados para teste de primalidade, um dos algoritmos é baseado no Crivo de Eratóstenes [2] e o outro é baseado na própria definição de números primos, que é pela contagem de divisores [3].

Na Figura 1a, é apresentado o algoritmo para teste de primalidade baseado no Crivo de Eratóstenes. Na Figura 1b, é apresentado o algoritmo para teste de primalidade baseado na contagem de divisores de um número. Ambos os algoritmos foram implementados em Python, utilizando os próprios recursos nativos da linguagem.

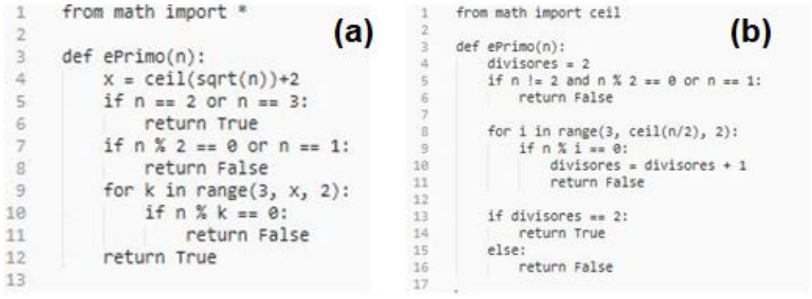

Figura 1: (a) Algoritmo Crivo de Eratóstenes. (b) Algoritmo contagem de divisores.

[1]david.vinicius@aluno.ifsp.edu.br
[2]nogueira.paulo@ifsp.edu.br

Para realização dos experimentos foi utilizado um computador com as seguintes especificações: Intel Core I7-3770, 32 GB de RAM, Linux CentOS 6.9.

A análise de desempenho dos algoritmos foi realizada comparando os tempos de execução de cada um em seis cenários diferentes. Cada um dos algoritmos tiveram valores de entrada variando do intervalo de 1 a 10^n, sendo n o cenário de teste. Os dois algoritmos foram executados 31 vezes para cada cenário de teste, sendo que a primeira replicação de cada cenário foi descartada para evitar a influência de *buffer-cache* de disco.

Os resultados foram tabulados e disposto nas Tabelas 1 e 2.

Tabela 1: Algoritmo por contagem de divisores - tempo em segundos

Até	Média	Mediana	Variância	Desvio Padrão	TMax	TMin
10	0,006	0,010	2,54E-05	0,050	0,010	0,000
100	0,005	0,010	2,575E-05	0,005	0,010	0,000
1000	0,008	0,010	1,655E-05	0,004	0,010	0,000
10.000	0,147	0,150	2,023E-05	0,004	0,150	0,140
100.000	12,060	12,045	0,0012654	0,036	12,200	12,030
1.000.000	1100,727	1100,420	4,3859651	2,094	1107,220	1098,230

Tabela 2: Algoritmo baseado no Crivo de Eratóstenes - tempo em segundos

Até	Média	Mediana	Variância	Desvio Padrão	TMax	TMin
10	0,006	0,010	2,483E-05	0,500	0,010	0,000
100	0,005	0,000	2,57E-05	0,500	0,010	0,000
1000	0,007	0,010	2,172E-05	0,500	0,010	0,000
10.000	0,019	0,020	6,437E-06	0,300	0,020	0,010
100.000	0,199	0,200	6,437E-06	0,300	0,200	0,190
1.000.000	4,024	4,020	4,875E-04	0,022	4,070	3,990

Como pode ser observado, houve uma discrepância entre os dois algoritmos conforme o aumento dos valores de entrada. Para valores até 1000, há diferenças insignificantes quando comparado os dois algoritmos. No entanto, nota-se que há uma grande diferença entre os tempos de execução dos algoritmos para valores acima de 10.000.

A média de tempo do algoritmo por contagem de divisores é maior que o algoritmo baseado no Crivo de Eratóstenes. Portanto, pode ser observado a superioridade em termos de desempenho do algoritmo do Crivo. Assim, com base nos resultados apresentados na Tabela 1 e Tabela 2 e considerando apenas o tempo de execução, caso seja necessário adotar um dos dois algoritmos para descobrir e atestar se um determinado número primo, fica evidente que o Crivo de Eratóstenes é a melhor opção para os números dentro dos intervalos 10^1 a 10^6.

Referências

[1] LEMOS, Manoel. *Criptografia, números primos e algoritmos*. s. Rio de Janeiro: IMPA, 2001.

[2] GOW, Mary. *Measuring the Earth: Eratosthenes and His Celestial Geometry*, Berkeley Heights, NJ: Enslow, 2010. p. 6

[3] RIESEL, Hans. *Prime Numbers and Computer Methods for Factorization*. 2 ed. Estocolmo:Birkhäuser, 1994.

[4] IMPA. *Descoberto número primo com quase 25 milhões de dígitos*. 2019. Disponível em: https://impa.br/noticias/descoberto-numero-primo-com-quase-25-milhoes-de-digitos/. Acesso em: 28 fev. 2021.