

Raízes primitivas módulo n

Gustavo Zambonin

Segurança em Computação (UFSC-INE5429)

- O escopo deste trabalho é discutir o conceito de *raízes primitivas módulo n* . Este construto matemático pode ser definido da seguinte forma: seja n um número inteiro; então g é uma raiz primitiva módulo n se, para cada inteiro a coprimo¹ a n , existe um inteiro $k \in \mathbb{Z}^*$ tal que $g^k \equiv a \pmod{n}$. Verifica-se, por exemplo, que 2 é uma raiz primitiva módulo 13:

$$\begin{array}{llllll} 2^0 = & 1 = & 2^0 \equiv & 1 = & 1 \equiv & 1 \pmod{13} \\ 2^1 = & 2 = & 2^1 \cdot 2 \equiv & 1 \cdot 2 = & 2 \equiv & 2 \pmod{13} \\ 2^2 = & 4 = & 2^2 \cdot 2 \equiv & 2 \cdot 2 = & 4 \equiv & 4 \pmod{13} \\ 2^3 = & 8 = & 2^3 \cdot 2 \equiv & 4 \cdot 2 = & 8 \equiv & 8 \pmod{13} \\ 2^4 = & 16 = & 2^4 \cdot 2 \equiv & 8 \cdot 2 = & 16 \equiv & 3 \pmod{13} \\ 2^5 = & 32 = & 2^5 \cdot 2 \equiv & 3 \cdot 2 = & 6 \equiv & 6 \pmod{13} \\ 2^6 = & 64 = & 2^6 \cdot 2 \equiv & 6 \cdot 2 = & 12 \equiv & 12 \pmod{13} \\ 2^7 = & 128 = & 2^7 \cdot 2 \equiv & 12 \cdot 2 = & 24 \equiv & 11 \pmod{13} \\ 2^8 = & 256 = & 2^8 \cdot 2 \equiv & 11 \cdot 2 = & 22 \equiv & 9 \pmod{13} \\ 2^9 = & 512 = & 2^9 \cdot 2 \equiv & 9 \cdot 2 = & 18 \equiv & 5 \pmod{13} \\ 2^{10} = & 1024 = & 2^{10} \cdot 2 \equiv & 5 \cdot 2 = & 10 \equiv & 10 \pmod{13} \\ 2^{11} = & 2048 = & 2^{11} \cdot 2 \equiv & 10 \cdot 2 = & 20 \equiv & 7 \pmod{13} \\ 2^{12} = & 4096 = & 2^{12} \cdot 2 \equiv & 7 \cdot 2 = & 14 \equiv & 1 \pmod{13} \end{array}$$

...

No caso de números n primos, as potências de g formam um ciclo que não pode ser maior do que $n - 1$ (pelo Pequeno Teorema de Fermat). Assim, g é uma raiz primitiva módulo n pois produz todos os *resíduos* possíveis módulo n . Nota-se que um número só possui esta característica caso a cardinalidade do conjunto destas potências seja igual a $\phi(n)$. Como um número primo s não tem divisores a não ser 1 e ele mesmo, então todos os números no intervalo $[1, s - 1]$ são coprimos a ele, logo $\phi(s) = s - 1$.

- A partir do conceito elaborado acima, é possível encontrar facilmente todas as raízes primitivas de um número se apenas uma já é conhecida. Seja g uma raiz primitiva módulo n , onde n é um primo ímpar (para simplicidade da demonstração). Então sabe-se que g pode gerar todas as potências da forma $g^m \equiv 1 \pmod{n}$, para $(m = 1 \dots n - 1)$. Para estes números serem raízes primitivas, $(a^m)^{(n-1)/d} \equiv (a^{n-1})^{m/d} \equiv 1 \pmod{n}$, então precisa-se de $d = 1$. O código em `primitive_root.py` explora uma variante dessa análise que utiliza-se da escolha de números aleatórios como “chutes” para a possível raiz primitiva (estes números devem estar dentro do grupo multiplicativo $\mathbb{Z}/n\mathbb{Z}$ ² para que a análise seja válida).

A partir da congruência $a^{(p-1)/q} \not\equiv 1 \pmod{p}$ para todos os fatores primos de $p - 1$, é possível verificar se a é uma raiz primitiva módulo n , e construir uma lista sucessivamente. A complexidade para achar uma raiz primitiva singular aleatoriamente funciona *relativamente bem*, porém a velocidade diminui exponencialmente em comparação com uma estratégia de iteração sobre todos os valores possíveis quando é necessário construir uma lista completa.

```
$ python
>>> import primitive_root as pr
>>> for i in [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]:
...     print("Primitive roots of {}: {}".format(i, pr.prim_roots(i, False)))
...
Primitive roots of 2: [1]
```

gustavo.zambonin@grad.ufsc.br

[src/](#)

¹números são coprimos, ou primos entre si, se seu único divisor positivo em comum é o número 1. A *função totiente de Euler*, $\phi(n)$, é responsável por fornecer a contagem de coprimos para um inteiro n .

²o conjunto de classes de congruência relativamente primos ao número.

Primitive roots of 3: [2]
 Primitive roots of 5: [2, 3]
 Primitive roots of 7: [3, 5]
 Primitive roots of 11: [2, 6, 7, 8]
 Primitive roots of 13: [2, 6, 7, 11]
 Primitive roots of 17: [3, 5, 6, 7, 10, 11, 12, 14]
 Primitive roots of 19: [2, 3, 10, 13, 14, 15]
 Primitive roots of 23: [5, 7, 10, 11, 14, 15, 17, 19, 20, 21]
 Primitive roots of 29: [2, 3, 8, 10, 11, 14, 15, 18, 19, 21, 26, 27]
 Primitive roots of 31: [3, 11, 12, 13, 17, 21, 22, 24]
 Primitive roots of 37: [2, 5, 13, 15, 17, 18, 19, 20, 22, 24, 32, 35]
 Primitive roots of 41: [6, 7, 11, 12, 13, 15, 17, 19, 22, 24, 26, 28, 29, 30, 34, 35]
 Primitive roots of 43: [3, 5, 12, 18, 19, 20, 26, 28, 29, 30, 33, 34]
 Primitive roots of 47: [5, 10, 11, 13, 15, 19, 20, 22, 23, 26, 29, 30, 31, 33, 35, 38, 39, 40, 41, 43, 44, 45]

- A aplicação mais comum para raízes primitivas módulo n ocorre no método assimétrico de troca de chaves chamado de Diffie-Hellman. Neste método, o número primo base e uma de suas raízes primitivas são utilizados para calcular a chave secreta final, resultado da aritmética modular proposta pelos autores. Este método é considerado seguro pois, para que este seja inviabilizado, é necessário resolver um problema chamado de *logaritmo discreto*: um inteiro k que resolve a equação $b^k = g$, onde b e g são elementos (não necessariamente números reais) cuidadosamente escolhidos. Métodos eficientes para a resolução deste problema não são conhecidos, e diversos algoritmos de criptografia assimétrica baseiam sua segurança nessa “dificuldade”.

Uma aplicação menos conhecida ocorre no design de difusores acústicos modulares [1]; um difusor com uma estrutura baseada em raízes primitivas possibilita a prevenção de reflexão de onda na direção especular³, assim absorvendo uma maior quantidade de som, neste caso.

Referências

- [1] R. Walker. *The Design and Application of Modular, Acoustic Diffusing Elements*. BBC R&D, London, UK, January 1990.

³reflexão de ondas similar a um espelho, onde uma onda é refletida para uma direção única, como num lago.