Algoritmo aleatorizado para detección de números primos

Alberto Benavides
21 de mayo de 2020

1. Introducción

Se sabe que la cantidad aproximada de números primos en los primeros n números naturales viene dada por la expresión $\frac{n}{\log n}$ y que, por el teorema de números primos de Lagrange, para un número de m dígitos existe una probabilidad de $1/m$ de ser primo. Una manera de determinar si un número x es primo consiste en calcular p mód $a, a = [2, \sqrt{x}], a \in \mathbb{N}$ de modo que x es primo si todos los módulos dan resultados distintos a cero, mientras que no sería primo si algún módulo da por resultado cero. Este método resulta lento si se desea saber si un número muy grande es primo. Es por esto que se puede optar por aproximaciones distintas para resolver este problema de decisión.

En relación con lo anterior, existen dos tipos de *algoritmos aleatorizados*,

- los algoritmos de Monte Carlo que obtienen soluciones con márgenes de error en poco tiempo mediante la aleatorización de valores de entrada, y
- los algoritmos de Las Vegas en que se obtiene un resultado exacto a costa de un tiempo de cómputo aleatorio que depende de iteraciones que se realizan de algoritmos de Monte Carlo.

Con la combinación de estas metodologías y tomando en cuenta el pequeño teorema de Fermat, que define que para un número primo p se cumple, para cada número $1 < a < p-1$ coprimo de p, que $a^{p-1} \equiv 1 \pmod{p}$, se puede realizar un algoritmo aleatorizado para determinar si p es primo que genere k números a al azar para probar la condición del pequeño teorema de Fermat, mismo que se muestra en el algoritmo 1. En este algoritmo

consiste la prueba de Miller-Rabin[1]. En este algoritmo, se le llama *testigos* a los números compuestos a que niegan las dos condiciones del algoritmo.

Algoritmo 1: Algoritmo aleatorizado para determinar si un número dado es primo.

para $i \in k$ **hacer**
  Elegir a aleatorio de $[2, p-1]$;
  si $m.c.d.(1, p) \neq 1$ **entonces**
    devolver $\perp$;
  fin
  si a^{p-1} mód $p \neq 1$ **entonces**
    devolver $\perp$;
  fin
fin
devolver $\top$

2. Probabilidad

Cada iteración de este algoritmo tiene tres posibles resultados:

- Si p es primo, siempre se retorna $\top$.
- Si p es compuesto y se retorna $\perp$, a es testigo de que p es compuesto.
- Si p es compuesto y se retorna $\top$. Esto es un falso positivo.

Interesa saber la probabilidad de veces en que el tercer caso sucedería. Se sabe que existen ciertos números compuestos que pasan la prueba de este algoritmo, llamados números Carmichael, como el 561. Experimentalmente se ha demostrado que estos números dan falsos positivos en la prueba de Miller-Rabin con una probabilidad $P(p \text{ es primo}) \leq 1/4$ de las ocasiones [1]. Por esto se puede concluir que después de i iteraciones en las que no se cumplan las condiciones de la prueba, la probabilidad $P(p \text{ es compuesto}) = (3/4)^i$.

Referencias

[1] Ben Lynn. Primality Tests. `https://crypto.stanford.edu/pbc/notes/numbertheory/millerrabin.html`, 2020. [Accedido 23/mayo/2020].

[2] Elisa Schaeffer. Complejidad computacional de problemas y el análisis y diseño de algoritmos. [No publicado], 2019.