Matematicas

Wilmer Emiro Castrillón Calderón

22 de agosto de 2022

1. MCD y MCM

MCD (Máximo común divisor)

El máximo común divisor de un conjunto de dos o mas números enteros es el máximo número entero el cual divide a todos los números del conjunto sin dejar residuo, por ejemplo el máximo común divisor de 35 y 15 es 5 porque es el máximo entero el cual los puede dividir a ambos. Como abreviatura se utiliza MCD o en ingles GCD (greatest common divisor).

El calculo del MCD de dos números se puede hacer de manera eficiente utilizando el algoritmo de Euclides el cual se puede realizar en $O(log(n))$ pasos, antes de explicarlo es necesario tener en cuenta las dos siguientes propiedades de la divisibilidad ($x|y$ significa que x divide a y con residuo 0).

$$x|y \rightarrow x|\alpha y \quad \forall \alpha \in Z \tag{1}$$

$$x|y \wedge x|y \pm z \rightarrow x|z \tag{2}$$

El algoritmo consiste en lo siguiente: dado dos números a y b se realiza la división entre ambos, obteniendo un cociente q y un residuo r, entonces $a = b * q_1 + r_1$ con $r_1 < b$, teniendo en cuenta que el MCD divide a a y b entonces también divide $b * q_1$ (propiedad 1), y también divide a r_1 (propiedad 2), el proceso se reduce a encontrar el MCD entre b y r_1 entonces se repite el proceso, $r_1 = b * q_2 + r_2$ con $r_2 < r_1$, y así sucesivamente hasta llegar a $r_n = 0$, lo cual indica que r_{n-1} divide a r_{n-1}, r_{n-2}, ..., b y a, entonces r_{n-1} es el MCD.

$$\begin{aligned} a &= b * q_1 + r_1 & r_1 &< b \\ b &= r_1 * q_2 + r_2 & r_2 &< r_1 \\ r_1 &= r_2 * q_3 + r_3 & r_3 &< r_2 \\ &... & & \\ r_{n-1} &= r_n * q_{n+1} + 0 & 0 &< r_n \end{aligned}$$

Por ejemplo, con $a = 35$ y $b = 15$, $a = b * 2 + 5$, $r_1 = 5$, en el siguiente paso $b = 5 * 3 + 0$, $r_2 = 0$, como se ha llegado a un residuo 0, el algoritmo finaliza y la respuesta es el ultimo residuo diferente de 0, entonces $MCD(35, 15) = 5$.

MCM (Mínimo común múltiplo)

El mínimo común múltiplo de un conjunto de números enteros es el numero entero mas pequeño el cual es múltiplo de todos los números del conjunto, por ejemplo el mínimo común múltiplo de 35 y 15 es 105, pues es el menor entero tal que 35|105 y 15|105. Como abreviatura se usa MCM o en ingles LCM (lowest common multiple).

Para calcular el MCM también se puede utilizar el algoritmo de Euclides, pues existe una relación entre el MCD y el MCM. entonces $MCM(a*b) = (a*b)/MCD(a,b)$, esta formula es equivalente a $a*(b/MCD(a,b))$ como $MCD(a,b)|b$ se puede observar que el resultado sera un entero múltiplo de a, de igual es equivalente a $b*(a/MCD(a,b))$ y el resultado es un entero múltiplo de b, entonces $(a*b)/MCD(a,b)$ es múltiplo común de a y b.

Por ejemplo, con $a = 35$ y $b = 15$ el $MCM(a,b) = 35*15/5$ (en el ejemplo anterior se calculo el MCD de a y b), entonces el resultado es 105.

Implementación

La implementación del MCD consiste en simplemente aplicar los pasos descritos para el algoritmo de Euclides. La implementación del MCM consiste en aplicar la formula, se recomienda realizar primero la división para evitar un posible overflow al trabajar con números grandes.

```
1 int MCD(int a, int b) {
2   if(b) return MCD(b, a % b);
3   return a;
4 }
```

MCD

```
1 int MCM(int a, int b) {
2   return a*(b/MCD(b, a % b));
3 }
```

MCM

2. Criba de Eratóstenes

La búsqueda de números primos es un problema clásico, las soluciones sencillas se basan en tomar un numero x y empezar a validar si algún numero en el intervalo $[2, x-1]$ divide a x, si ninguno lo divide entonces x es un numero primo, esta idea se puede mejorar un poco, se puede validar desde el comienzo si x es un numero par, en ese caso x es primo solamente si es igual a 2, pues el único numero primo par es el 2, si x es impar se hace la validación si algún numero impar en el intervalo $[3, x-1]$ divide a x, existe una propiedad la cual indica que el menor factor primo de un numero x es menor a $\sqrt{x}$, entonces nuevamente es posible reducir el intervalo, finalmente un entero positivo x mayor a 1 es primo si es igual a 2 o si ningún numero impar en el intervalo $[3, \sqrt{x}]$ divide a x. Este método tiene complejidad $O(\sqrt{x})$ para cada numero, si se necesita verificar n números, el proceso se tendría que repetir n veces, por lo tanto si se requiere obtener una lista de números primos se tiene que buscar una mejor solución

La criba de Eratóstenes es un algoritmo para encontrar todos los números primos hasta un numero n, cosiste de un arreglo de números los cuales se irán marcando si son compuestos, si no están marcados entonces son primos, inicialmente todos son considerados primos (no están marcados), se comienza tomando el 2 y se marcan todos los múltiplos de 2 como números compuestos, después se toma el 3 y se marcan todos los múltiplos de 3 como números compuestos, al llegar al 4 este ya estará marcado entonces el 4 es compuesto, en este caso se omite y se pasa al 5 el cual no esta marcado, y así sucesivamente, los números que no estén marcados son

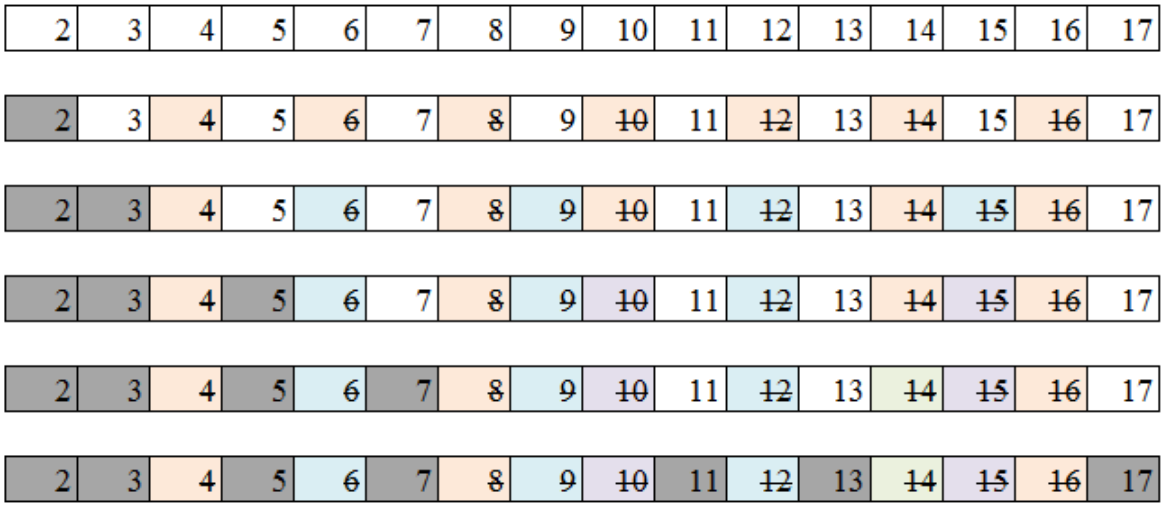

Figura 1

números primos, en la figura 1 se puede observar un sencillo ejemplo con $n = 17$.

La implementación consiste en usar un arreglo de bool inicialmente en true (a excepción del 0 y 1 que no son primos), después se recorre el arreglo verificando los números que no estén marcados, para cada numero i no marcado se empieza a marcar los múltiplos de i, los múltiplos se pueden empezar a marcar desde i^2. Al final se obtiene un arreglo el cual permite validar si un i es primo y permite obtener un vector con la lista de números primos hasta n, el algoritmo tiene complejidad $O(n * log(log(n)))$.

```
bool esPrimo[1000005];
vector<int> primos;

void criba(int n) {
    memset(esPrimo, true, sizeof(esPrimo));
    esPrimo[0] = esPrimo[1] = false;

    for(int i = 2; i < n; ++i){
        if(!esPrimo[i]) continue;

        for(int j = i*i; j < n; j += i)
            esPrimo[j] = false;
        primos.push_back(i);
    }
}
```