Name: Taher Mohamed Section: 1 BN: 39

1- $\gcd(ac, bc) = c.\gcd(a, b)$

let $d = \gcd(a, b)$

then $\rightarrow d = xa + yb$

$cd = x(ca) + y(cb) \rightarrow$ ①

$\because d|a$ and $d|b$ then $d|ca$ and $d|cb$ (properites) $\rightarrow$ ②

From ①,② $\gcd(ac, bc) = c.\gcd(a, b)$

2- If $\gcd(a, c) = 1$, then $\gcd(a, bc) = \gcd(a, b)$

let $d = \gcd(a, bc)$ and $b \neq 1$

$\because d|a$ and $\gcd(a, c) = 1 \rightarrow d \nmid c \rightarrow$ ①

$\because d|bc$ and $d \nmid c \rightarrow d|b \rightarrow \rightarrow$ ②

$\because d = x(a) + y(bc) = xa + ycb = xa + y' b \rightarrow$ ③

From ①,②,③ $\rightarrow d = \gcd(a, b)$

3- $\gcd(a, b) = \gcd(a-b, b)$ assuming $a > b$

let $d = \gcd(a, b)$

$\because d|a$ and $d|b \rightarrow a = cd$ and $b = rd$

$\rightarrow a-b = d(c-r)$

$\rightarrow d|(a-b) \rightarrow$ ①

$\because d = ax + by = ax + by - bx + bx$

$= x(a-b) + (y+x)b \rightarrow$ ②

From ① ② $d = \gcd(a-b, b) = \gcd(a, b)$

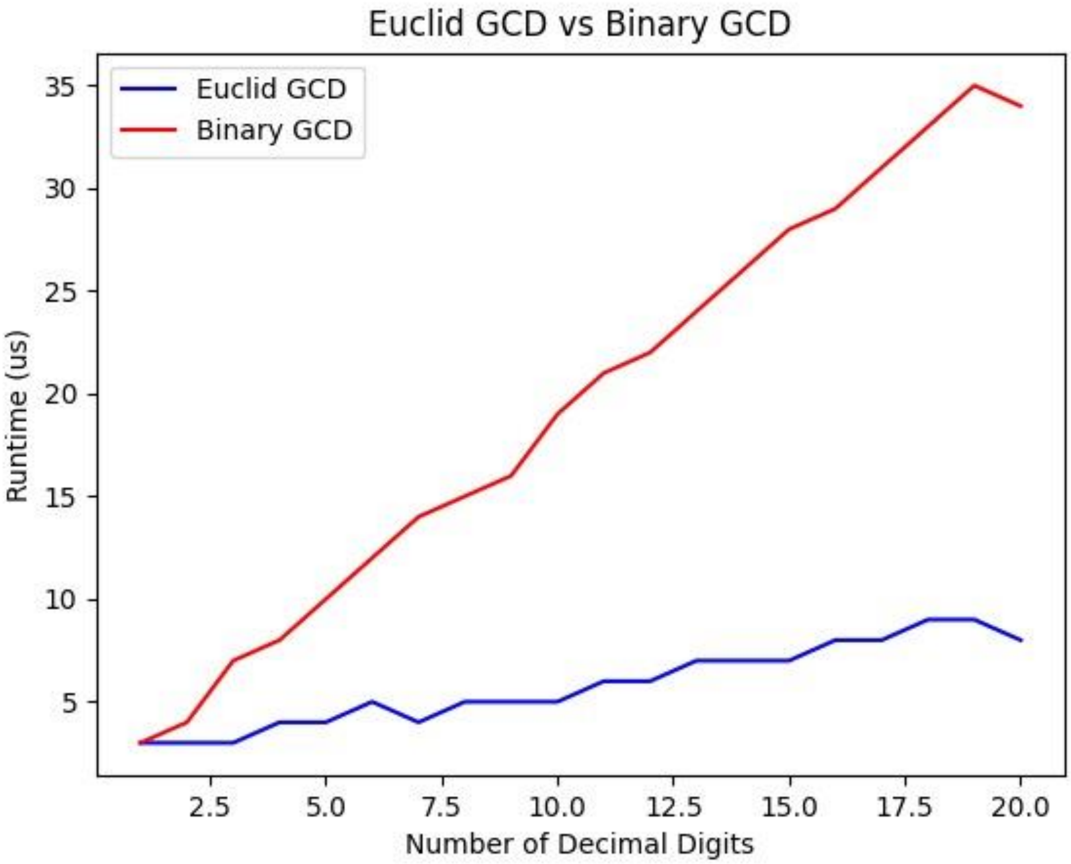
Intuition

This algorithm replaces division by 2 using shift right and replaces multiplying by 2 using shift left. The division operation has high cost in CPU so this algorithm is better than Euclid in most cases. In some cases, the Euclid is faster the Binary GCD and this is machine dependent and depends on the compiler.

If a and b are even then we can use the first statement we proved to get gcd (a, b) = 2 gcd (a/2, b/2).

If one of them is odd assume it is a, we can use the second statement proved to conclude that gcd (a, b) = gcd (a, b/2).

If both are odd, we can use the third statement which proved to conclude that gcd (a, b) = gcd (a-b, b).


Euclid GCD vs Binary GCD
Euclid GCD
Binary GCD
Runtime (us)
Number of Decimal Digits