Álgebra Abstracta (CCOMP3-1)

Nombres: Juan Pablo Galindo Coronel

2021-2 Laboratorio 02b

01/10/2021: — Prof. Jose Chavez

Theorem 1. *Si $a > b \geq 1$, donde $a, b \in \mathbb{N}$ y $\texttt{EUCLID}(a, b)$ requiere $k \geq 1$ pasos para obtener el máximo común divisor. Entonces $a \geq F_{k+2}$ y $b \geq F_{k+1}$.*

Proof. La demostración procede por inducción.

- Base: Asumimos que $k = 1$. Entonces

$$b \geq 1 = F_2 = F_{1+1} \quad (k = 1),$$

y además $a \geq b + 1 \geq 2 = F_3 = F_{1+2} \quad (k = 1)$.

- Paso inductivo: Suponemos que el teorema se cumple si se requieren $n \geq 1$ pasos. Entonces debemos probar que el teorema se mantiene para $n + 1$.

Como $\texttt{EUCLID}(a, b)$ llama a $\texttt{EUCLID}(b, a \mod b)$ de manera recursiva, entonces este último realizará n pasos para calcular el máximo común divisor. Utilizando la hipótesis inductiva se tiene que $b \geq F_{n+2}$ (aquí se prueba parte del teorema) y $(a \mod b) \geq F_{n+1}$. Si sumamos ambas desigualdades obtenemos lo siguiente

$$\begin{aligned} b + (a \mod b) &\geq F_{n+2} + F_{n+1} \\ b + (a - \left\lfloor \frac{a}{b} \right\rfloor b) &\geq F_{n+3} \\ a - b(\left\lfloor \frac{a}{b} \right\rfloor - 1) &\geq F_{n+3} \\ a \geq a - b(\left\lfloor \frac{a}{b} \right\rfloor - 1) &\geq F_{n+3} \qquad (\left\lfloor \frac{a}{b} \right\rfloor \geq 1) \end{aligned}$$

Al final se tiene que $a \geq F_{n+3}$.

1. (5 points) Calcular el tiempo computacional del Algoritmo de Euclides. Detalle y sustente su respuesta.

Algoritmo de Euclides es un método eficiente para encontrar el MCD (mayor divisor común) de dos números enteros. La complejidad temporal de este algoritmo es

O (log (min (a, b))

Supongamos que ayb son dos números enteros tales que **a> b** entonces de acuerdo con el algoritmo de Euclides:

mcd (a, b) = mcd (b, a% b)

*Utilice la fórmula donde **b** es **0** . En este paso, el resultado será el **MCD de los dos enteros** , que será igual a **a** . Entonces, después de observar cuidadosamente, se puede decir que la complejidad temporal de este algoritmo sería proporcional al número de pasos necesarios para reducir **b** a*

0 *.Asumamos que el número de pasos necesarios para reducir* ***b*** *a* ***0*** *usando este algoritmo es* ***N***

`mcd (a, b)= N veces`

Ahora,si el euclidiana Algoritmo para dos números **a** y **b** se reduce en **N** pasos entonces, **una** debe ser al menos **f (N + 2)** y **b** debe ser al menos **f (N + 1)**.

mcd (a, b) ——> N pasos
Entonces, a> = f (N + 2) yb> = f (N + 1)
donde, f N es el (N- ésimo) término en la serie de Fibonacci (0, 1, 1, 2, 3,…) y N > = 0.
Para probar la afirmación anterior utilizando el principio de inducción matemática (PMI)

Caso Base

- Supongamos que **a = 2** y **b = 1** . Entonces, **mcd (2, 1)** se reducirá a **mcd (1, 0)** en **1** paso, es decir, **N = 1** .
- Esto significa que **2** debe ser **al menos f 3** y **1** debe ser **al menos f 2** y **f 3 = 2** y **f 2 = 1** .
- Esto implica que **a** es al menos **f (N + 2)** y **b** es al menos **f (N + 1)** .
- Se puede concluir que la afirmación es cierta para el caso base.
- **Paso inductivo:** Supongamos que la afirmación es válida para la **(N - 1) º Paso** . Entonces, a continuación se muestran los pasos para
  demostrar **N- ésimo paso**
  mcd (b, a% b) ——> (N - 1) pasos
  Entonces,

  b> = f (N - 1 + 2) es decir, b> = f (N + 1)
  a% b> = f (N - 1 + 1) es decir, a% b> = f

- *Ahora, (a / b) siempre sería mayor que 1. Entonces, del resultado anterior, se concluye que*

  a> = b + (a% b)

  Esto implica, a> = f (N + 1) + f N

  *se puede decir que si el algoritmo de Euclides para dos números **a** y **b** se reduce en **N pasos** entonces, **una** debe ser al menos f **(N + 1)***

se concluye que:

=> f N + 1 ≈ min (a, b)

=> N + 1 ≈ log ⌀ min (a, b)

=> O (N) = O (N + 1) = log (min (a, b))

2. (15 points) Implementar el Algoritmo Extendido de Euclides.

   - Implementar un programa que permita ingresar dos números a y b (enteros positivos) y que retorne $\{\texttt{gcd}(a,b), x, y\}$, donde $\texttt{gcd}(a,b) = ax + by$ $(x, y \in \mathbb{Z})$.
   - Enviar un enlace al repositorio (única forma de envío). En este colocará un `README.md` con una breve descripción del programa y las instrucciones para ejecutarlo. Colocar un ejemplo del resultado esperado.

https://github.com/juanpa022/Extendido-de-Euclides

funcionamiento:

En este código podemos observar que mientras nuestra while y sea verdadero va a seguir cumpliendo nuestras funciones tal caso nosotros tenemos primero tres variables que son respectivamente a lo que nosotros pide como máximo común divisor luego tendremos dos números mas para que nos ayude con los cambios, que van a ser los números que sacaremos el MDC luego tenemos nuestra división en esta resta por y luego está multiplicación cambiamos de variables actualizando A y B luego tengo que actualizar a s1 y s2 ,tendremos nuestro resultado el cual solo nos da un número por lo

que en mi main he tenido que reutilizar otra vez mi código es un defecto que he encontrado pero no significa que el código no esté bien.

```cpp
#include <iostream>
using namespace std;
int euclidesExt(int b,int a)
{
    int x,y,r;
    int s1 = 0, s2 = 1;
    while(b>0){
        x = a/b;
        r = a - x*b;
        a = b;
        b = r;
        y = s1 - (x*s2);
        s1 = s2;
        s2 = y;
    }
    return s1;
}
int main()
{
  cout<<euclidesExt(1554,366)<<"\n";
  cout<<euclidesExt(366,1554);
  return 0;
}
```

```
clang++-7 -pthread -std=c++17 -o main main.cpp
./main
-4
17
```