Algoritmos y Complejidad (INF-221)
Tarea 1
Algoritmo de Strassen
v/s
Algoritmo Tradicional de Multiplicación de Matrices

Rodrigo Cayazaya - 201773538-4
Francisco Reyes - 201773529-5
Lucio Fondón - 201773610-0

3 de Julio de 2020

1 Introducción

En este informe se realizarán comparaciones de tiempo de ejecución para dos algoritmos de multiplicación de matrices; el algoritmo de **Strassen** ($O(n^{2,807})$) y el algoritmo tradicional ($O(n^3)$), con la finalidad de ver el mejor desempeño. Para esto se realizarán varias pruebas con inputs de matrices de tamaño n x n, en donde $n = 2^m$ con $m \in \mathbb{N}$.

Nota: El valor de la constante k siempre será 1, debido a que el algoritmo de Strassen tendría desventaja contra el algoritmo tradicional, ya que tendría que calcular una matriz cuadrada de potencia de 2 más grande que la original y el algoritmo tradicional utilizaría la original.

2 Realización de pruebas de ejecución

Para poder realizar las comparaciones de los dos algoritmos, necesitamos los tiempos promedio de ejecución de cada uno de éstos. Para obtener una buena estimación de cuánto se demora cada algoritmo, se probará con un total de 20 ejecuciones para cada uno y se calculará el promedio del total de éstas ejecuciones. La siguiente tabla contiene los tiempos de ejecución promedio de cada algoritmo con su respectivo tamaño n de la matriz cuadrada:

Matriz n	Strassen $t_{strassen}[s]$	Tradicional $t_{tradicional}[s]$
8	2,04E-03	1,47E-04
16	1,29E-02	8,83E-04
32	9,94E-02	6,11E-03
64	5,81E-01	4,18E-02
128	4,10E+00	3,15E-01
256	2,98E+01	2,45E+00
512	2,05E+02	2,01E+01

Tabla 1: Tiempos de ejecución promedio para n = 2^m, con $m = 3, 4, \ldots, 9$.

Notemos que los tiempos de ejecución son siempre mayores en al menos un orden de magnitud para el algoritmo de Strassen. Además, cabe destacar que si seguimos probando con $n > 512$, empezará a aumentar considerablemente el orden de magnitud de los tiempos de ejecución para cada algoritmo y tomará mucho tiempo computar éstas soluciones.

3 Estimación Lineal

Dado lo anterior, procederemos a hacer una estimación lineal con los datos obtenidos, para así poder encontrar funciones lineales aproximadas que generen los tiempos de ejecución para cada algoritmo dado un n. Como nota, se realizará una estimación lineal aplicando el logaritmo en base de 10 ($\log_{10} n$) a todos los datos, esto para poder ver una linealidad en los datos graficados.

```
Coeficientes estimados de  Strassen:
Intercepto = -5.206142302112073

Pendiente = 2.770134680108295


Coeficientes estimados tradicional:
Intercepto = -6.471186597684138

Pendiente= 2.8481163029460226
```

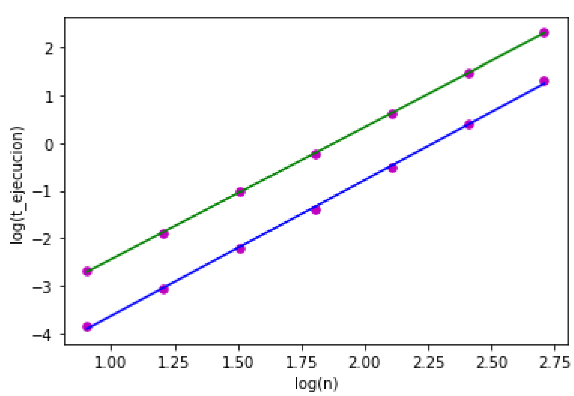

Gráfico 1: Estimaciones lineales para Strassen (verde) y tradicional (azul)

Notemos que la pendiente de la estimación lineal para el algoritmo de Strassen es ligeramente menor a la pendiente del algoritmo tradicional, por lo que, en algún punto dado (en este caso algún $\log_{10} n$) se encontrarán estas funciones y Strassen podrá tener un tiempo menor o igual al tiempo del algoritmo tradicional.

Procedemos a calcular este intercepto igualando las funciones, para así poder obtener el $\log_{10} n$ en común y finalmente, elevar este valor a la potencia de 10 para así obtener el verdadero n necesario para que Strassen tengo un tiempo menor o igual al del algoritmo tradicional, es decir:

$$t_{strassen} = m_{strassen} \cdot \log_{10} n + b_{strassen} \tag{1}$$
$$t_{tradicional} = m_{tradicional} \cdot \log_{10} n + b_{tradicional} \tag{2}$$

$$t_{strassen} = t_{tradicional} \tag{3}$$

Despejando $\log_{10} n$ obtenemos:

$$\log_{10} n = \frac{b_{strassen} - b_{tradicional}}{m_{strassen} - m_{tradicional}} \tag{4}$$

Reemplazando los valores obtenidos anteriormente en la estimación lineal, obtenemos que:

$$\log_{10} n = 16,22228207 \tag{5}$$

Lo que nos da un valor de $n = 1,67E + 16$, es decir, $t_{strassen} \leq t_{tradicional}$ solo para n de orden de magnitud ≥ 16.

4 Conclusiones

A lo largo del desarrollo de este escrito y de los experimentos, hemos llegado a la conclusión de que el orden del algoritmo Strassen es menor por muy poco al algoritmo tradicional de multiplicación de matrices. Esto lo podemos ver a través de una estimación lineal de ambas funciones.

Para nuestro caso en particular, Strassen tiene mejor desempeño *ssi* las dimensiones de la matriz son extremadamente altas (n de orden de magnitud ≥ 16). Sin embargo, este valor n pudo haberse visto afectado por la implementación que se realizó del algoritmo de Strassen y también por la elección del lenguaje en el que se realizó el código, el cuál fue Python, y como bien sabemos, es un lenguaje de alto nivel y tiene muchas subrutinas que realiza sin que el programador las realice explícitamente. En cambio, si la implementación se hubiese realizado en lenguajes de más bajo nivel ya sea C/C++, tal vez y muy probablemente las rectas de la estimación lineal estarían muchos más cerca una de la otra, y por ende, el intercepto entre estas dos funciones sería un n de orden de magnitud mucho más pequeño.