UNIVERSIDAD TECNICA FEDERICO SANTA MARIA | DEPARTAMENTO DE INFORMÁTICA

Análisis del algoritmo de Strassen para la multiplicación de matrices

Joaquín Ignacio Calderón Donoso - joaquin.calderond@usm.cl - 201973571-3

Daniel Alejandro Martínez Castro - daniel.martinezc@usm.cl - 201973508-k

Tomás Santiago Nieto Guerrero - tomas.nieto@usm.cl - 201973529-2

1 Resumen

En este informe mostraremos y analizaremos los resultados de dos algoritmos diferentes para la multiplicación de dos matrices de largo 2^n. En particular compararemos el algoritmo de multiplicación de matrices tradicional con el algoritmo de Strassen, siendo este último nuestra principal motivación para desarrollar este experimento. Pudimos evidenciar la diferencia de complejidad de los algoritmos bajo estudio.

2 Introducción

En el año 1969 Strassen publica por primera vez su algoritmo para la multiplicación de matrices $n * n$, demostrando tener una complejidad de $\Theta(n^{2.81})$, menor al algoritmo tradicional con complejidad de $\Theta(n^3)$. Para lograr esto, Strassen se basó en el conocido diseño de algoritmos llamado dividir-y-conquistar.

Dividir-y-conquistar rompe los problemas a los que se enfrenta, recursivamente, en sub-problemas menos complejos, pero del mismo tipo. El proceso de romper los problemas en problemas más fáciles se hace hasta que el sub-problema se vuelve lo suficiente sencillo para resolverlo directamente. Para conocer la solución al problema original, debemos unir todas las soluciones de los sub-problemas creados.

Conceptualmente, el algoritmo de Strassen divide la matriz cuadrada en cuatro sub-matrices, donde sigue dividiéndose recursivamente hasta que la sub-matriz sea de $1 * 1$, es decir, de un elemento. Al llegar al caso base de este problema, se hacen las operaciones definidas por el algoritmo de strassen, para luego unir la solución, escalar en los sub-problemas, hasta poder retornar la multiplicación de las matrices originales.

El algoritmo de Strassen se puede hacer de manera más eficiente si en vez de llegar al caso base de matrices de $1 * 1$ se llega hasta un N donde el algoritmo de multiplicación tradicional es más eficiente que el de Strassen. Esto suele ocurrir alrededor de matrices de largo $N = 100$, pero depende mucho del hardware donde se ejecute este algoritmo. Para nuestro caso escogimos arbitrariamente un $N = 128$ de tal forma que cuando se reducen las matrices hasta $128 * 128$ se usa el algoritmo tradicional y termina la recursión.

Para comparar experimentalmente al algoritmo de Strassen y el tradicional usaremos un programa escrito en c^{++} que ejecutará estos algoritmos para un tamaño de matriz dado, y luego retornará el tiempo que se empleó para cada solución.

3 Método para la extracción de datos

Como habíamos dicho en el apartado introductorio, c^{++} será el lenguaje que usaremos para implementar los dos algoritmos bajo estudio. Para conocer los tiempos de ejecución usamos la función now() de la librería crono. Analizaremos en este caso 12 tamaños del problema de multiplicación de matrices cuadradas. En específico, estas matrices cuadradas serán de largo $2^n, n \in Z^+ / 1 \leq n \leq 12$.

Los datos se tomaron ejecutando el programa con una CPU AMD Ryzen 5 3400G.

El código utilizado se encuentra en el repositorio Github.

4 Resultados

A continuación, se presenta un gráfico que muestra el tiempo de ejecución de cada algoritmo, dado un exponente, cabe decir, el exponente correspondiente al tamaño de la matriz $2^n * 2^n$.

Gráfico 1: Tiempo de ejecución en función de n. Algoritmos tradicional y de Strassen en Azul y Naranja, respectivamente.

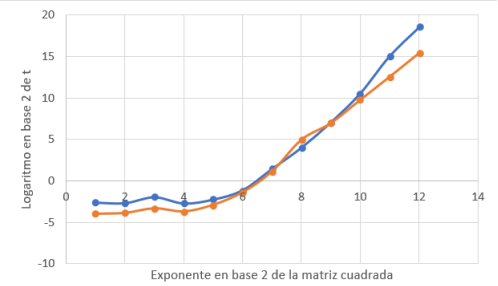

Gráfico 2: Tiempo de ejecución normalizado en función de n. Algoritmo tradicional y de Strassen en Azul y Naranja, respectivamente.

Como podemos ver la diferencia en los tiempos se nota a lo largo de todos los tamaños de las matrices (ver Anexo 1). Esto se aprecia más fácilmente en el gráfico 2, ya que los tiempos de ejecución están normalizados. En cuanto al gráfico 1, podemos notar que el algoritmo tradicional evidentemente crece mucho más rápido en tiempo de ejecución. Esto comprueba nuestra observación hecha en los apartados anteriores, donde mencionamos la diferencia de complejidad de los dos algoritmos bajo estudio.

Además podemos notar en el gráfico 2, que en el valor $n = 8$ el tiempo del algoritmo de Strassen es mayor que el algoritmo tradicional, esto se debe al valor elegido para terminar la recursión. En nuestro caso este valor es de $N = 128$, el cual como se menciono anteriormente fue elegido arbitrariamente. Si se escoge un valor apropiado para N esto no ocurriria y Strassen tendría menor tiempo de ejecución.

Conclusión

El algoritmo de Strassen es conocido por haber reducido la complejidad del problema de multiplicar matrices cuadradas, haciendo uso de dividir-y-conquistar. En esta experiencia pudimos analizar los dos algoritmos de interés, implementándolos en c^{++}, evidenciando las diferencias de tiempo empleado.

En conclusión, en esta experiencia pudimos estudiar un algoritmo que hace uso de dividir-y-conquistar, logrando ver una aplicación real de este diseño de algoritmos y su relevancia.

Anexos

n	tradicional [ms]	strassen [ms]
1	0,163	0,064
2	0,156	0,068
3	0,257	0,100
4	0,152	0,077
5	0,211	0,135
6	0,457	0,385
7	2,722	2,191
8	16,009	31,019
9	133,117	124,378
10	1.485,370	863,769
11	33.526,300	5.982,230
12	373.918,000	43.195,500

Anexo 1: Tiempos de ejecución para multiplicación de matrices de $2^n * 2^n$ usando el algoritmo tradicional y el de Strassen.