Tarea 1

Algoritmos y Estructuras de Datos Avanzados / Magister en Cs. de la Computación

Respuestas:

1. Usando las ideas anteriores, generar al azar las matrices A y B (considere matrices de enteros) y completar las siguiente tabla con los tiempos de ejecución1. DR1 usa la primera propiedad, y DR2 usa la segunda (prográmelos en el lenguaje que estime conveniente).

Para la resolución del punto 1, se utilizó el lenguaje de programación **Python**, con el cual se implementaron el algoritmo tradicional, y los algoritmos identificados como DR1 y DR2 dentro del enunciado.

Esta implementación se trata de un programa que permite al usuario definir un número entero "n", con el cual genera, de manera automática, dos matrices cuadradas de n $\times$ n con valores aleatorios. Luego, estas dos matrices son multiplicadas a través de los tres algoritmos ya mencionados. Finalmente, el tiempo de ejecución de cada uno de ellos es mostrado por pantalla para poder compararlos en la tabla a continuación.

	Tiempos		
n	**Algoritmo Tradicional**	**DR1**	**DR2**
32	30 ms	0 ms	0 ms
64	242 ms	0 ms	0 ms
128	1925 ms	2 ms	1 ms
256	15347 ms	21 ms	18 ms
512	124653 ms	267 ms	165 ms
1024	1002550 ms	6928 ms	3002 ms
2048	N/A	N/A	N/A
4060	N/A	N/A	N/A

Para efectos de esta tarea, solo se han considerado valores para $n \leq 1024$, ya que los tiempos de espera son razonables dentro de ese rango.

2. Obtenga al menos dos conclusiones, respecto del rendimiento de los algoritmos.

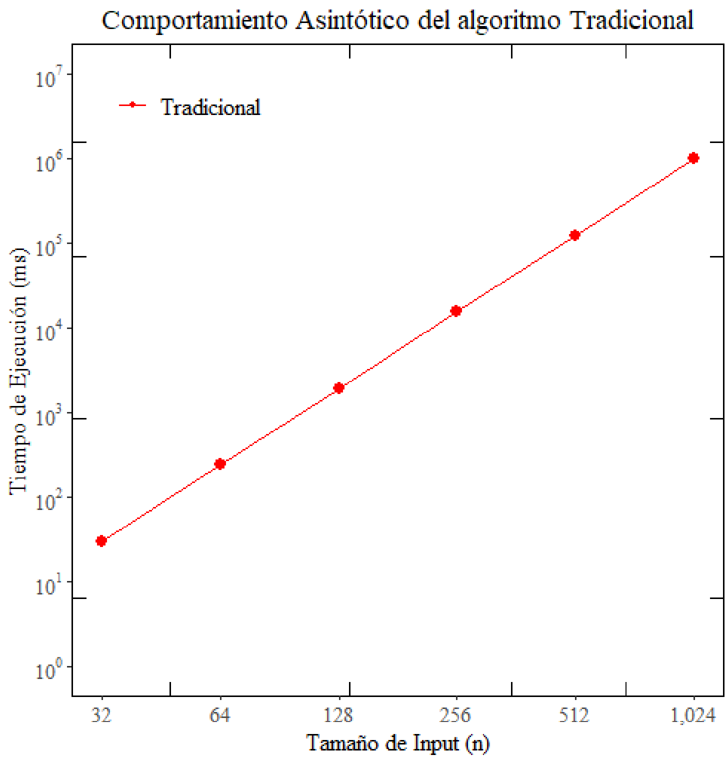
- **Conclusión 1**: A medida que el orden de las matrices (n) aumenta, el rendimiento del algoritmo tradicional se vuelve notablemente peor que el de los algoritmos DR1 y DR2.

- **Conclusión 2**: Los algoritmos DR1 y DR2 escalan mucho mejor que el algoritmo tradicional, ya que sus tiempos de ejecución crecen mucho más lento a medida que aumenta el orden de las matrices (n). Gracias a esto, DR1 y DR2 son mucho mejores para trabajar con matrices de orden mayor.

3. Haga un estudio de comportamiento asintótico de los 2 algoritmos que creó.

En este punto se presentan una serie de gráficos que ilustran la relación entre el tamaño del input (n) y tiempo de ejecución para los algoritmos implementados, junto con una breve explicación de su implementación y características.

En primer lugar, tenemos el algoritmo tradicional, el cual posee una complejidad de $O(n^3)$.



En segundo lugar, tenemos el algoritmo DR1, el cual se trata del "Divide y Conquista", o "Divide and Conquer". Este algoritmo introduce el concepto de subdividir las matrices que se están multiplicando en matrices más pequeñas y multiplicar estas en lugar de aplicar una multiplicación bruta elemento por elemento, como lo hace el algoritmo tradicional. En el enunciado podemos ver lo siguiente, en donde A y B son las matrices que se están multiplicando, y C la matriz producto.

$$C11 = A_{11} \cdot B_{11} + A_{12} \cdot B_{21}$$
$$C12 = A_{11} \cdot B_{12} + A_{12} \cdot B_{22}$$
$$C21 = A_{21} \cdot B_{11} + A_{22} \cdot B_{21}$$
$$C22 = A_{21} \cdot B_{12} + A_{22} \cdot B_{22}$$

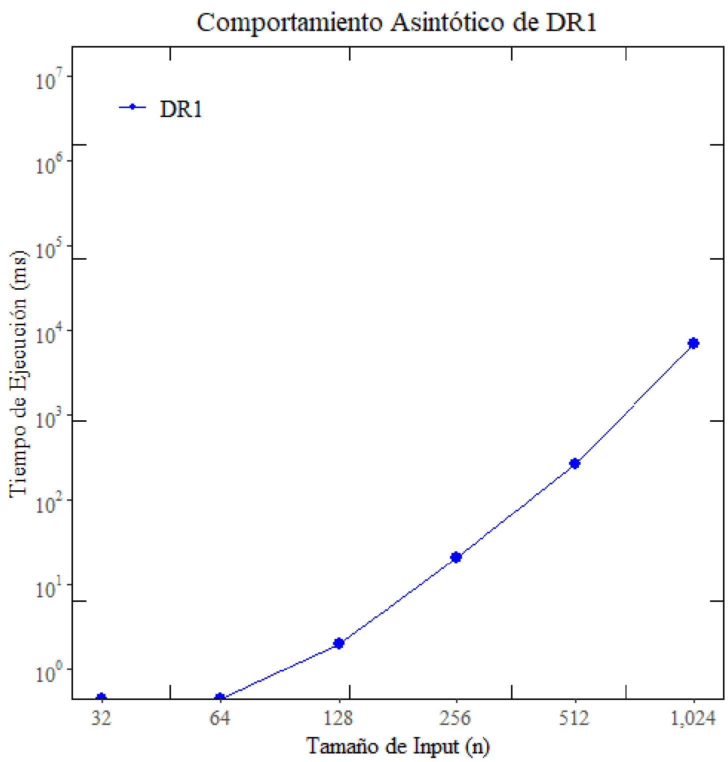
Si sabemos que la suma de las matrices es de complejidad $O(n^2)$, por lo que la complejidad temporal puede escribirse como:

$$T(n) = 8T(n/2) + O(n^2)$$

Aplicando el Teorema Maestro Simple en su tercera opción, obtenemos que:

$$T(n) = O(n^{\log_2 8})$$
$$T(n) = O(n^3)$$

La complejidad de DR1 es parecida a la del algoritmo tradicional, pero se debe tener en cuenta que esta es la expresión más básica del algoritmo de divide y conquista para este propósito.



Por último, DR2 representa la implementación del algoritmo de Strassen, el cual lleva un paso más adelante la estrategia de divide y conquista de la siguiente manera:

$$M = (A_{11} + A_{22})(B_{11} + B_{22})$$
$$N = (A_{21} + A_{22})B_{11}$$
$$O = A_{11}(B_{12} - B_{22})$$
$$P = A_{22}(B_{21} - B_{11})$$
$$Q = (A_{11} + A_{12})B_{22}$$
$$R = (A_{21} - A_{11})(B_{11} + B_{12})$$
$$S = (A_{12} - A_{22})(B_{21} + B_{22})$$
$$C_{11} = M + P - Q + S$$
$$C_{12} = O + Q$$
$$C_{21} = N + P$$
$$C_{22} = M + O - N + R$$

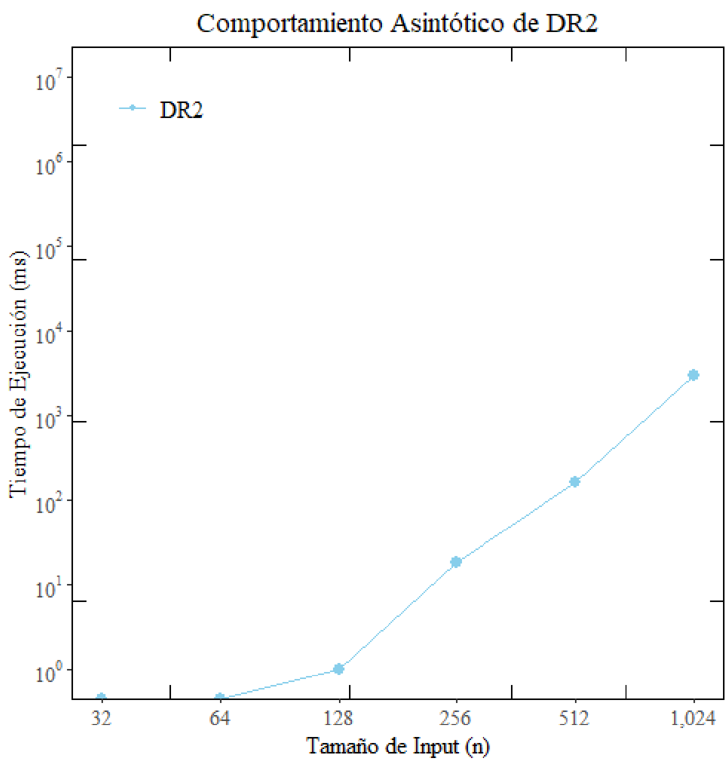
El algoritmo de Strassen también divide las matrices, pero éste elimina una de las llamadas recursivas, quedadon únicamente en 7.

La complejidad temporal de este algoritmo es entonces:

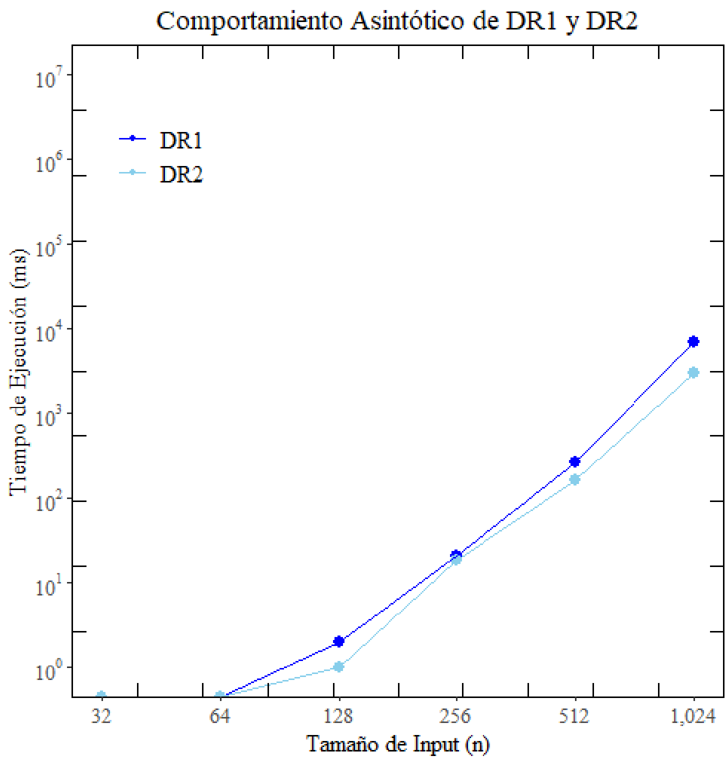
$$T(n) = 7T(n/2) + O(n^2)$$

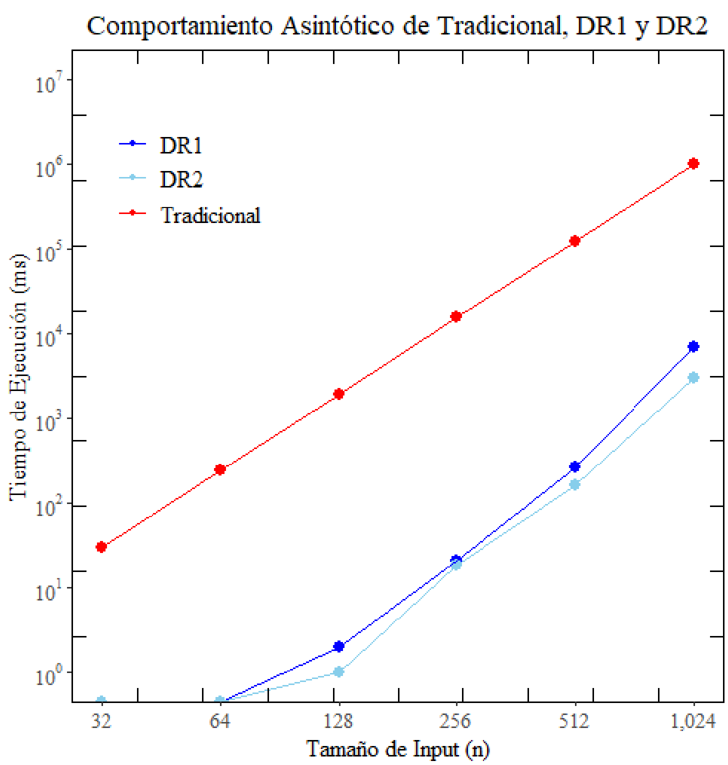
Por Teorema Maestro Simple, opción 3:

$$O(n^{Log7}) \approx O(n^{2.8074})$$



Para finalizar, se muestra una vista comparativa entre DR1 y DR2, y en segundo lugar una comparación general de esos dos con el algoritmo tradicional.


Comportamiento Asintótico de DR1 y DR2
DR1
DR2
Tiempo de Ejecución (ms)
Tamaño de Input (n)
32
64
128
256
512
1,024


Comportamiento Asintótico de Tradicional, DR1 y DR2
DR1
DR2
Tradicional
Tiempo de Ejecución (ms)
Tamaño de Input (n)
32
64
128
256
512
1,024