**Assignment 3 (Question 1)**

Analysis for Strassen’s algorithm:

|  |  |  |
| --- | --- | --- |
|  | **Brute Force Time (nanoseconds)** | **Strassen’s Time (nanoseconds)** |
| 2 | 6998 | 133429 |
| 4 | 21460 | 452536 |
| 8 | 120832 | 3059987 |
| 16 | 847224 | 11716033 |
| 32 | 5974892 | 70860701 |
| 64 | 17348505 | 457900799 |
| 128 | 56112752 | 2001382336 |
| 256 | 168924688 | 17073630038 |
| 512 | 974610795 | 112435712976 |
| 1024 | 11805961362 | 757510201434 |
| 2048 | 231504494115 |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | **Brute Force Time (milliseconds)** | **Strassen’s Time (milliseconds)** |
| 100 | 46496.102 | 682474.656 |
| 200 | 66276.645 | 2362983.811 |
| 300 | 411491.180 | 8291133.595 |
| 400 | 566887.803 | 13198730.285 |
| 500 | 894061.032 | 12953978.427 |
| 600 | 1211243.451 | 75948285.796 |
| 700 | 1546133.606 | 69914723.442 |
| 800 | 3089495.537 | 69395735.798 |
| 900 | 3860478.104 | 105698394.148 |
| 1000 | 8507026.457 | 98903282.197 |

Analysis about Strassen’s algorithm:

As shown in the graph, if we use the array to build the Strassen’s algorithm, it takes quite long to execute the matrix multiplication.

Time complexity for Strassen’s Algorithm is O(n2.81) whereas that of Brute force Algorithm is O(n3). But the constant that comes along the power of n might be so big to overcome the normal multiplication value.

If we use Linked List for building Strassen’s algorithm, it shows efficiency while doing it rather than using arrays for it. The test cases shows significance difference.

|  |  |  |
| --- | --- | --- |
|  | **Brute Force** | **Strassen’s Using Linked List** |
| 4 | 0 | 0 |
| 8 | 0 | 2 |
| 16 | 0 | 2 |
| 32 | 1 | 8 |
| 64 | 1 | 10 |
| 128 | 10 | 24 |
| 256 | 120 | 84 |
| 512 | 621 | 512 |
| 1024 | 33010 | 4334 |
| 2048 | 470539 | 12289 |

After empirical analysis using linked list, we can conclude that after matrix size = 128, Strassen’s algorithm works more efficiently than the traditional brute force.

This is because, in traditional method we use arrays to perform matrix computations which has a time complexity of O(n). Whereas, linked list insertion can be computed in the O(1) time. This results in a considerable time difference when matrix size is large.