

算法设计与分析实验报告

实验名称：矩阵乘法(分治法)

一、问题陈述，相关背景、应用及研究现状的综述分析

1.问题陈述：

设 A 和 B 是两个 $n * n$ 阶矩阵，求它们的乘积矩阵 C 。这里，
假设 n 是 2 的幂次方

2.相关背景：

1969 年，Volker Strassen 发表文章提出一种渐进快于平凡算法的矩阵相乘算法，引起巨大轰动。在此之前，很少人敢设想一个算法能渐进快于平凡算法。矩阵乘法的渐进上界自此被改进了。

$$\begin{bmatrix} r & s \\ t & u \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \cdot \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

$$P_1 = a \cdot (f - h)$$

$$P_2 = (a + b) \cdot h$$

$$P_3 = (c + d) \cdot e$$

$$P_4 = d \cdot (g - e)$$

$$P_5 = (a + d) \cdot (e + h)$$

$$P_6 = (b - d) \cdot (g + h)$$

$$P_7 = (a - c) \cdot (e + f)$$

$$r = P_5 + P_4 - P_2 + P_6$$

$$s = P_1 + P_2$$

$$t = P_3 + P_4$$

$$u = P_5 + P_1 - P_3 - P_7$$

7次乘法！！

18次加法/减法

二、模型拟制、算法设计和正确性证明

1. 算法设计:

利用 C++ 类定义一个矩阵类 `Matrix`, 然后对于矩阵类重载 `+`, `-`, `*`, `=` 运算符来方便计算。然后定义 `void Strassen(Matrix& A, Matrix& B, Matrix& C)` 函数, 在其中将矩阵 `A, B, C` 分块, 计算对应的 `A11, A12, A21, A22, B11, B12, B21, B22`, 进一步利用递归计算 `M1` 到 `M7`, 然后计算 `C11, C12, C21, C22`, 最后合并成为结果 `C`, 递归出口为最简单的 2 维矩阵相乘, 直接调用普通矩阵乘法运算。

三、时间和空间复杂性分析

时间复杂度：

$$T(n) = 7T(n/2) + \Theta(n^2)$$

$$T(n) = \Theta(n^{\lg 7}) \approx \Theta(n^{2.81})$$

空间复杂度：

每次对一个矩阵的分块都会使用四个新的矩阵，新矩阵的维数是原矩阵的一半，但是四个新的矩阵的大小和原矩阵是一样的，所以消耗的新空间和原矩阵的维数有关，设矩阵维数是 2 的 m 次幂，则

$$S(n) = O(m \cdot 3n^2) = O(n^2)$$

四、程序实现和实验测试过程

源程序见 Strassen_Matrix_Multiplication.cpp

测试过程：

测试 8 维矩阵，矩阵内容由随机数定义。

```
C:\Users\xx\Desktop\Strassen_Matrix_Multiplication\bin\Debug\S
1 2 4 0 4 4 3 3
2 4 0 0 1 2 1 1
0 2 2 1 1 4 2 3
2 2 1 1 3 0 2 1
1 3 4 2 2 4 0 4
3 1 2 3 3 4 1 1
3 3 2 4 2 2 2 4
3 1 4 3 1 0 0 2

3 1 0 2 4 3 1 0
0 4 0 0 1 1 3 3
4 3 4 4 1 0 1 1
1 3 4 4 1 3 2 3
3 2 4 1 3 0 4 3
4 0 2 1 3 1 1 0
4 2 0 4 2 3 2 2
1 0 1 1 0 2 0 4

62 35 43 41 40 24 37 40
22 22 9 12 23 17 22 21
39 23 27 28 24 21 22 30
29 26 21 24 25 19 27 27
47 35 44 36 31 24 30 41
50 30 41 38 41 28 32 29
47 41 40 46 37 40 36 49
33 30 34 37 23 23 20 27
```

五、总结

对矩阵类重载运算符可以使矩阵运算清晰方便许多。