**（1）4.2-2 为strassen算法编写伪代码**

**Strassen(A,B)**

**n=A.rows**

**let C be a new n\*n matrix**

**if n==1**

**c=a\*b**

**else partition A,B,and C as in equations({A = [A11 A12 ; A21 A22] , B = [B11 B12 ; B21 B22] , C = [C11 C12 ; C21 C22]})**

**S1=B12-B22**

**S2=A11-A12**

**S3=A21+A22**

**S4=B21-B11**

**S5=A11+A22**

**S6=B11+B22**

**S7=A12-A22**

**S8=B21+B22**

**S9=A11-A21**

**S10=B11+B12**

**P1=Strassen(A11,S1)**

**P2=Strassen(A11,B22)**

**P3=Strassen(S3,B11)**

**P4=Strassen(A22,S4)**

**P5=Strassen(S5,S6)**

**P6=Strassen(S7,S8)**

**P7=Strassen(S9,S10)**

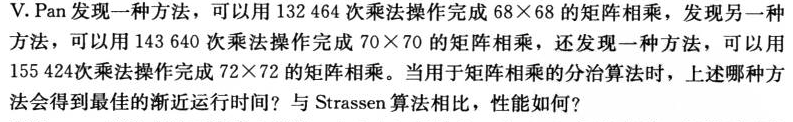
**C11=P5+P4-P2+P6**

**C12=P1+P2**

**C21=P3+P4**

**C22=P5+P1-P3-P7**

**return C**

1. **4.2-5**

log68132464≈2.795128<lg7

log70143640≈2.795122<lg7

log72155424≈2.795147<lg7

所以，就题目给出的条件来看，第二种方法会得到最佳的渐进运行时间，且这三种方法性能略优于Strassen算法。

**（3）证明三数取中选择Pivot, 划分出来的子数组的比例平均值为5：11**

**（4）快速查找法，如果每次划分的比例不差于1：19，给出最坏时间复杂度的递推方程，并求解**