算法第三次作业

王道宇 `PB21030794`

4.2-3

因为 n 不为 2 的整数次幂，故必然存在 $k \in Z^+$，使得 $2^k < n < 2^{k+1}$，将 n 矩阵补充成 2^{k+1} 阶，空余位置补 0 ，类似：

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix}$$

此时矩阵相乘的结果并不会改变。由于：

$$\Theta((2^{k+1})^{log(7)}) = \Theta(7 \cdot (2^k)^{log(7)}) = \Theta((2^k)^{log(7)})$$

所以，该算法的时间复杂度仍然为 $\Theta(n^{log(7)})$。

4.2-5

用于矩阵的分治乘法时，当 n 足够大时，可以将矩阵分成 68×68 份（或其他两种方法），这样三种方法的递推公式分别是：

$$T_1(n) = 132464 \cdot T_1(\frac{n}{68}) + \Theta(n^2) \tag{1}$$

$$T_2(n) = 143640 \cdot T_2(\frac{n}{70}) + \Theta(n^2) \tag{2}$$

$$T_3(n) = 155424 \cdot T_1(\frac{n}{72}) + \Theta(n^2) \tag{3}$$

由主方法知，三种方法的 $T(n)$ 分别可以写成：

$$T_1(n) = \Theta(n^{log_{68}(132464)}) = \Theta(n^{2.795128}) \tag{1}$$

$$T_2(n) = \Theta(n^{log_{70}(143640)}) = \Theta(n^{2.795122}) \tag{2}$$

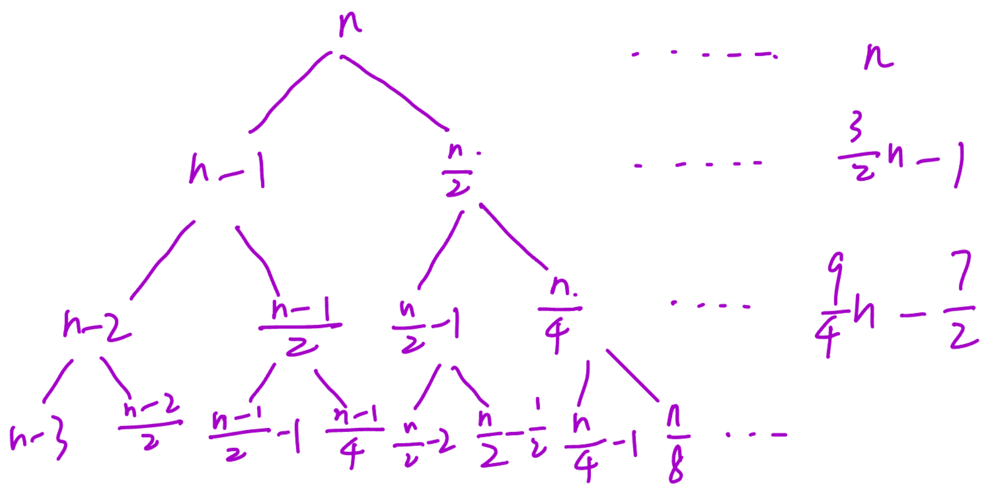
$$T_3(n) = \Theta(n^{log_{72}(155424)}) = \Theta(n^{2.795147}) \tag{3}$$

可以看到在应用到分治算法之后 $T_2(n)$ 的渐进运行时间最佳，同时，三种方法都优于 $Strassen$ 算法，因为 $Strassen$ 算法的 $T(n) = \Theta(n^{log(7)}) = \Theta(n^{2.807355})$。

4.4-5

递归式 $T(n)=T(n-1)+T(\frac{n}{2})+n$。

递归树如下:



设顶层为第 1 层，下方以此类推，设每层的子问题所产生的代价为 $g(k)$，其中 $k=1$时，$g(1)=n$，由于最长路径为 $n\to(n-1)\to(n-2)\to\cdots\to 1$，故递归树一共 n层。

有递推公式:

$$g(k)=\frac{3}{2}g(k-1)-2^{k-2} \tag{1}$$

由 (1)式可做如下推导:

$$\frac{g(k)}{2^{k-2}}+4=\frac{3}{4}(\frac{g(k-1)}{2^{k-3}}+4) \tag{2}$$

$$g(k)=-2^k+(n+2)(\frac{3}{2})^{k-1} \tag{3}$$

然而，这个递归公式在 $k>log(n)$就不适用了，同样的，在某个 n_0 之后，$g(k)$将会小于 0。

所以考虑两种极端情况:

- 若 $T(n)=2T(n-1)+n$，那么 $T(n)=O(2^n)$
- 若 $T(n)=2T(\frac{n}{2})+n$，那么由主方法可得 $T(n)=O(nlog(n))$

可以确定地本题中 $T(n)=\Omega(nlog(n))$，并且 $T(n)=O(2^n)$，还发现，$\forall a\geq 1+\epsilon$，其中 ϵ 为任意正数，总有：$T(n)=o(a^n)$，因为当 n 足够大的时候，总有 $a^{n/2}>\frac{2(1+\epsilon)}{\epsilon}$和 $a^n>\frac{2(1+\epsilon)}{\epsilon}n$，对于任意 $c>0$ 均有:

$$\begin{aligned}
T(n)&=T(n-1)+T(\frac{n}{2})+n\\
&<ca^{n-1}+ca^{n/2}+n\\
&=ca^n-ca^n(1-\frac{1}{a}-\frac{1}{a^{n/2}}-\frac{n}{a^n})\\
&\leq ca^n-ca^n(\frac{\epsilon}{2(1+\epsilon)}-\frac{1}{a^{n/2}}+\frac{\epsilon}{2(1+\epsilon)}-\frac{n}{a^n})\\
&<ca^n
\end{aligned}$$

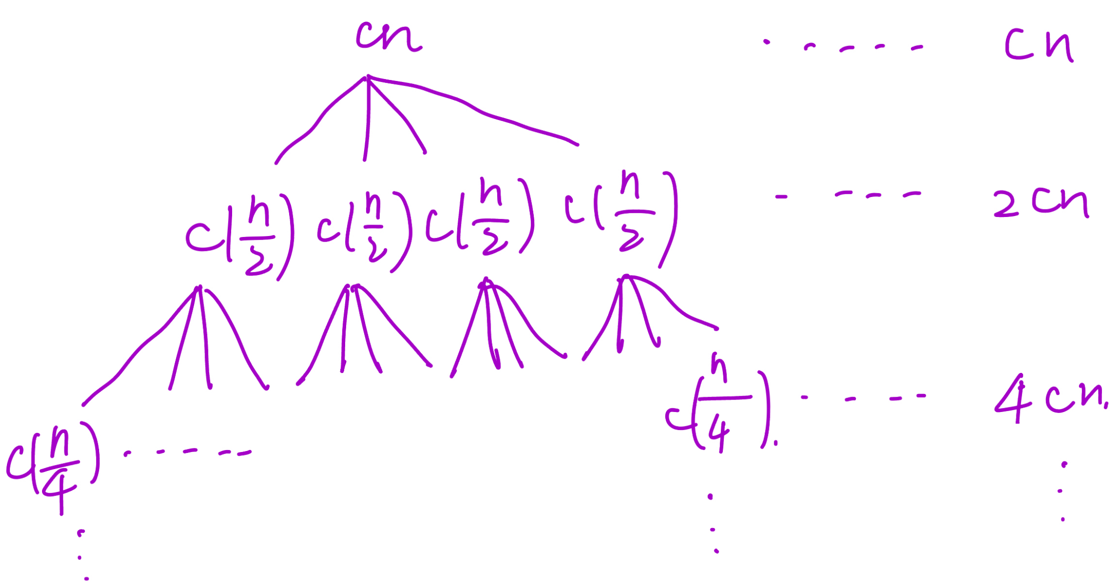
同理其实可以证明 $T=\omega(n^klog^m(n))$，其中 k, m为任意正数。

鉴于本体只让我们找一个较好的渐进上界，其实只要令 $a=\frac{3}{2}$，$T(n)=(\frac{3}{2})^n$ 即可。

4.4-7

递归式：$T(n)=4T(\lfloor\frac{n}{2}\rfloor)+cn$

递归树如下：



递归树一共 $log(n)+1$ 层，将每层子问题的代价求和：

$$
\begin{aligned}
T(n) &= \sum_{k=1}^{log(n)+1} 2^k cn \\
&= cn\sum_{k=1}^{log(n)+1} 2^k \\
&= cn(2^{log(n)+2}-2) \\
&= 4cn^2-2cn
\end{aligned}
$$

可以看出 $T(n)=O(n^2)$。

使用代入法进行验证：

由 $T(0)=0,\ T(1)=c$，取 $d=max\{c+g,\ \}$，$n\geq n_0=1$，其中 $g>\frac{c}{2}$：

首先有 $T(1)\leq d-g$。假设 $m<n$时，均有 $T(n)\leq dn^2-gn$，特别是 $T(\frac{n}{2})\leq c(\frac{n}{2})^2-g(\frac{n}{2})$，那么有：

$$
\begin{aligned}
T(n) &= 4T(\lfloor\frac{n}{2}\rfloor)+cn \\
&\leq 4T(\frac{n}{2})+cn \\
&\leq 4c(\frac{n}{2})^2-4g(\frac{n}{2})+cn \\
&= cn^2-2gn+cn \\
&\leq cn^2
\end{aligned}
$$

归纳结束，可以证明 $T(n)=O(n^2)$，并且是渐进紧确界。

4.5-2

由矩阵乘法的限制性可知，该递推公式必然要满足 `case1`。

由主方法，该递推公式的解是：

$$T(n)=\Theta(n^{log_4(a)})$$

要超过 $Stassen$ 方法，需要有：

$$log_4(a)<log_2(7)$$

解得：

$$a<49$$

故 a 的最大整数值应是 48

4.5.4

在递归式 $T(n)=4T(\frac{n}{2})+n^2log(n)$中：$a=4,\ b=2,\ f(n)=n^2log(n)$

分水岭函数为：$n^{log_ba}=n^2$

现证明其不能使用主方法求解：

- 由于 $n^2=o(n^2log(n))$，故 `case1` 不可能。
- 由于 $\lim\limits_{n\to+\infty}\frac{n^2}{f(n)}=\lim\limits_{n\to+\infty}\frac{1}{log(n)}=0$，故$n^2$与 $f(n)$不同阶，`case2` 不可能。
- 由于 $\forall\epsilon>0,\ n^\epsilon=\omega(log(n))$，故有 $n^{2+\epsilon}=n^2n^\epsilon=\Omega(n^2log(n))$，`case3` 不可能。

这个递归式的渐进上界是 $n^2log^2(n)$，即 $T(n)=O(n^2log^2(n))$，现证明此结论。

即要证：$\exists c,\ n_0>0,\ s.t.\ \forall n\geq n_0,\ 0\leq T(n)\leq cn^2log^2(n)$

取 $c=max\{\frac{T(2)}{4},\ 1\},\ n\geq n_0=2$:

首先有 $T(2)\leq c\cdot 2^2\cdot log^2(2)$。假设 $m<n$时，均有 $T(n)\leq cn^2log^2(n)$，特别是 $T(\frac{n}{2})\leq c(\frac{n}{2})^2log^2(\frac{n}{2})$，那么有：

$$\begin{aligned}T(n)&=4c(\frac{n}{2})^2log^2(\frac{n}{2})+n^2log(n)\\&=cn^2[log(n)-1]^2+n^2log(n)\\&=cn^2log^2(n)+(1-2c+\frac{c}{log(n)})n^2log(n)\\&\leq cn^2log^2(n)+(1-2c+\frac{c}{log(n_0)})n^2log(n)\\&=cn^2log^2(n)+(1-c)n^2log(n)\\&\leq cn^2log^2(n)\end{aligned}$$

归纳结束，可以证明 $T(n)=O(n^2log^2(n))$