非完美最近点对算法时间复杂度的证明

Shuxin Chen, 2015013229

2017 年 3 月 3 日

1. 已知 $T(n) = 2T(n/2) + \Theta(n \log n)$，求 $T(n)$。

   我们用递归的方法求解。

$$\begin{aligned} T(n) &= 2T(\frac{n}{2}) + O(n \log n) \\ &= \sum_{i=1}^{\lfloor \log n \rfloor} (2^{i-1} * \frac{n}{2^{i-1}} \log \frac{n}{2^{i-1}}) \\ &= \sum_{i=1}^{\lfloor \log n \rfloor} (n \log \frac{n}{2^{i-1}}) \\ &= n \sum_{i=1}^{\lfloor \log n \rfloor} (\log \frac{n}{2^{i-1}}) \end{aligned}$$

   令 $m = \lfloor \log n \rfloor, u_i = \log \frac{n}{2^{i-1}}$，我们有

$$T(n) = n \sum_{i=1}^{m} u_i$$

   我们发现，$u_i - u_{i-1} = -\log 2$，因此 $\{u_n\}$ 为等差数列，我们可以用求和公式

$$S_n = nu_n - \frac{n(n-1)}{2} d$$

   来计算数列的和，其中 d 为公差。

   设 $u(m) = \Theta(1)$，我们有

$$T(n) = n \sum_{i=1}^{m} u_i$$

$$
\begin{aligned}
&= n(mu_m - \frac{m(m-1)}{2} * (-\log 2)) \\
&= n(m\Theta(1) + \frac{m(m-1)}{2} * (\log 2)) \\
&= \Theta(nm^2) \\
&= \Theta(n \log^2 n)
\end{aligned}
$$

因此，$T(n) = \Theta(n \log^2 n)$。