11 月 3 日作业

韩岳成 524531910029

2025 年 11 月 3 日

题目 1. 请用主定理方法和递归树方法分别分析以下递归表达式的时间复杂度:

$$T(n) = 8T(\frac{n}{2}) + 100n^3$$

解答.

- 主定理方法:
  子问题个数 $a = 8$, 问题大小缩减因子 $b = 2$，$f(n) = 100n^3$。
  计算 $\log_b a = \log_2 8 = 3$, $f(n) = \Theta(n^3) = \Theta(n^{\log_b a} \log^0 n)$，符合主定理第二种情况，因此

$$T(n) = \Theta(n^{\log_b a} \log^{k+1} n) = \Theta(n^3 \log n)$$

- 递归树方法:
  递归树的高度为 $L = \log_2 n$,
  将根结点视为第 0 层，第 n 层拥有 8^n 个节点，每个节点的规模为 $\frac{n}{2^n}$，每个节点的代价为 $100(\frac{n}{2^n})^3 = \frac{100n^3}{8^n}$。
  因此第 $n(0 \le n \le L-1)$ 层的总代价为 $8^n \cdot \frac{100n^3}{8^n} = 100n^3$。
  叶子结点层每个节点的规模为 1，共有 n^3 个叶子结点，因此叶子结点

的总代价为 $\Theta(n^3)$。

因此总代价为

$$T(n) = 100n^3 \cdot \log_2 n + \Theta(n^3) = \Theta(n^3 \log n)$$

题目 2. 标准的矩阵乘法算法需要 $O(n^3)$ 的时间。Strassen 矩阵乘法是一种更高效的分治算法。它将一个 $n \times n$ 的大矩阵分解为若干个 $n/2 \times n/2$ 的子矩阵，巧妙地通过 7 次子矩阵的递归乘法和一系列的矩阵加减法（时间复杂度为 $O(n^2)$）来完成计算，减少了递归调用的次数。

Strassen 矩阵乘法的时间复杂度可以用以下递归表达式描述，请用主定理方法和递归树方法分别分析其时间复杂度。

$$T(n) = 7T(\frac{n}{2}) + O(n^2)$$

解答.

- 主定理方法：

  子问题个数 $a = 7$, 问题大小缩减因子 $b = 2$，$f(n) = O(n^2)$。

  计算 $\log_b a = \log_2 7 \approx 2.81$，$f(n) = O(n^2) = O(n^{\log_b a - \epsilon})$，符合主定理第一种情况，因此

  $$T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_2 7})$$

- 递归树方法：

  递归树的高度为 $L = \log_2 n$，

  将根结点视为第 0 层，第 i 层拥有 7^i 个节点，每个节点的规模为 $\frac{n}{2^i}$，每个节点的代价为 $O((\frac{n}{2^i})^2) = O(\frac{n^2}{4^i})$。

  因此第 $i(0 \le i \le L-1)$ 层的总代价为 $7^i \cdot O(\frac{n^2}{4^i}) = O(n^2(\frac{7}{4})^i)$.

  树高约为 $\log_2 n$，取 $i = \log_2 n$，叶子结点数为 $7^{\log_2 n} = n^{\log_2 7}$，每个叶

子结点的代价为 $\Theta(1)$，因此叶子结点的总代价为 $\Theta(n^{\log_2 7})$。因此总代价为

$$\begin{aligned}
T(n) &= \sum_{i=0}^{\log_2 n-1} O(n^2(\frac{7}{4})^i) + \Theta(n^{\log_2 7}) \\
&= O\left(n^2 \cdot \frac{(\frac{7}{4})^{\log_2 n} - 1}{\frac{7}{4} - 1}\right) + \Theta(n^{\log_2 7}) \\
&= O(n^{\log_2 4} \cdot n^{\log_2 \frac{7}{4}}) + \Theta(n^{\log_2 7}) \\
&= O(n^{\log_2 7}) + \Theta(n^{\log_2 7}) \\
&= \Theta(n^{\log_2 7})
\end{aligned}$$