ACM做题笔记

郝俊禹*

Contents

*Email:haojunyu2012@gmail.com

1 知识点总结

1.1 复杂度

复杂度比较常见的函数复杂度比较：

$$c < \log n < n < n \log n < n^d < a^n < n!$$

递归算法的时间复杂的情况有点复杂，其主定理由该公式计算：

$$T(n) = aT(\frac{n}{b}) + f(n)$$

a,b都是常数，n是规模变量，一般$f(n) = \mathcal{O}(n^d)$，d也是常数。经过推理1，可得一下结论：

- 若$d < \log_b^a$，$T(n) = \mathcal{O}(n^d)$
- 若$d = \log_b^a$，$T(n) = \mathcal{O}(n^d \log n)$
- 若$d > \log_b^a$，$T(n) = \mathcal{O}(n^{\log_b^a})$

附粗糙推理过程：假定$b^t = n$,即$t = \log_b^n$。

$$\begin{align}
T(n) &= aT(\frac{n}{b}) + f(n) \tag{1} \\
&= a^2T(\frac{n}{b^2}) + af(\frac{n}{b}) + f(n) \tag{2} \\
&= a^tT(1) + \sum_{i=0}^{t-1} a^i b^{(t-i)*d} \tag{3} \\
&= a^tT(1) + \frac{b^d}{b^d - a}(b^{dt} - a^t) \tag{4} \\
&= [T(1) - \frac{b^d}{b^d - a}]n^{\log_b^a} + \frac{b^d}{b^d - a}n^d \tag{5}
\end{align}$$

- 若$d < \log_b^a$，则等式5$= \mathcal{O}(n^d)$。
- 若$d = \log_b^a$，则等式3$= a^t(t + T(1)) = n^{\log_b^a}(\log_b^n + T(1)) = \mathcal{O}(n^d \log n)$。
- 若$d > \log_b^a$，则等式5$= \mathcal{O}(n^{\log_b^a})$。