《算法设计与分析》 Homework 4
Asymptotic Notation and Recurrences

January 8, 2027

Professor: Example Instructor

Ada Lovelace
2027000001

Problem 1

Give an appropriate positive constant c such that $f(n) \le c \cdot g(n)$ for all $n > 1$.

1. $f(n) = n^2 + n + 1$, $g(n) = 2n^3$
2. $f(n) = n\sqrt{n} + n^2$, $g(n) = n^2$
3. $f(n) = n^2 - n + 1$, $g(n) = n^2/2$

Solution

1. For $n > 1$: $n^2 + n + 1 \le n^2 + n^2 + n^2 = 3n^2 < 2n^3$ when $n > 1$, so $c = 1$ works for all $n > 1$.
2. $n\sqrt{n} + n^2 \le n^2 + n^2 = 2n^2$ for $n \ge 1$, so $c = 2$ works.
3. $n^2 - n + 1 \le n^2$ for $n \ge 1$, and $n^2 \le 2 \cdot (n^2/2)$, so $c = 2$ works.

Problem 2

Solve the recurrence $T(n) = 2T(n/2) + n$ using the recursion-tree method, and verify your answer using the Master Theorem.

Solution

At depth i there are 2^i subproblems, each of size $n/2^i$, so the total work at depth i is $2^i \cdot (n/2^i) = n$. The tree has $\log_2 n$ levels (down to subproblems of size 1), so

$$T(n) = \sum_{i=0}^{\log_2 n} n = n(\log_2 n + 1) = \Theta(n \log n). \tag{1}$$

By the Master Theorem with $a = 2$, $b = 2$, $f(n) = n$: since $n^{\log_b a} = n^{\log_2 2} = n = f(n)$, this is Case 2, giving $T(n) = \Theta(n \log n)$, matching the recursion-tree result.

Problem 3

Sample pseudocode listing (demonstrates the monospace font used for code blocks):

Algorithm 1 MergeSort(A, p, r)

```
1: if p < r then
2:     q ← ⌊(p + r)/2⌋
3:     MergeSort(A, p, q)
4:     MergeSort(A, q + 1, r)
5:     Merge(A, p, q, r)
6: end if
```

Solution

The recurrence for merge sort is $T(n) = 2T(n/2) + \Theta(n)$, which by the same Master Theorem case as Problem 2 gives $T(n) = \Theta(n \log n)$.