Quiz 1C

CS/CE 412/471 Algorithms: Design and Analysis, Spring 2025

27 Jan, 2025. 3 questions, 35 points, 5 printed sides

Instructions:

1. Enter your name and ID below and at the top of every side.
2. You have 60 minutes to complete this quiz.
3. You may keep a calculator with you. Submit all other devices with your bag at the front of the classroom.
4. Solve the problems by hand in clear and legible handwriting on your own paper.
5. Provide precise and concise solutions.
6. Consulting or copying from another student constitutes a violation of academic integrity. Any infractions will result in disciplinary action, including a failing grade for the quiz.
7. For your solution to be graded, submit this problem sheet along with your solution.

Student Name: _____

Student ID: _____

viel Spaß!

1. Algorithm Analysis and Correctness

(a) [10 points] Consider the following pseudocode to reverse an array of n elements:

```
ReverseArray(A, n)
1. for i = 1 to n/2
2.     temp = A[i]
3.     A[i] = A[n - i + 1]
4.     A[n - i + 1] = temp
5. return A
```

Prove the correctness of this algorithm using the loop invariant method.

Solution: We use the following loop invariant:

> At the start of each iteration of the loop, the first $i-1$ elements of the array have been reversed.

Proof. The algorithm is correct.

1. **Initialization:** Before the first iteration, no elements have been swapped, so the invariant holds trivially.
2. **Maintenance:** During each iteration, the algorithm swaps the i-th element with its counterpart at $n-i+1$. This operation ensures that the reversed property is extended to the next pair, maintaining the invariant.
3. **Termination:** At the end of the loop, $i = n/2$, and all pairs have been reversed. The entire array is therefore reversed.

□

(b) [5 points] Analyze its running time in terms of n, where n is the size of the array. Provide the best-case, worst-case, and average-case time complexities.

Solution:

Proof. The time complexity of each case is $\Theta\left(\frac{n}{2}\right) = \Theta(n)$.

1. The number of iterations, $\frac{n}{2}$, is independent of the data.
2. A constant amount of work takes place in each iteration.

□

2. Asymptotic Notation

(a) [5 points] Arrange the following functions in increasing order of growth rate. Justify your answer.

$$f_1(n) = \log n,\ f_2(n) = n,\ f_3(n) = e^n,\ f_4(n) = n^3.$$

Solution: Order: $f_1(n) = \log n < f_2(n) = n < f_4(n) = n^3 < f_3(n) = e^n$.

Proof. To compare the growth rates of these functions, we analyze the ratio of their growth rates as $n \to \infty$ using the following limit test:

$$\lim_{n\to\infty} \frac{f_i(n)}{f_j(n)}.$$

1. Compare $f_1(n) = \log n$ and $f_2(n) = n$:

   $$\lim_{n\to\infty} \frac{\log n}{n} = 0.$$

   Since the limit is 0, $f_1(n) = \log n$ grows slower than $f_2(n) = n$.

2. Compare $f_2(n) = n$ and $f_4(n) = n^3$:

   $$\lim_{n\to\infty} \frac{n}{n^3} = \lim_{n\to\infty} \frac{1}{n^2} = 0.$$

   Since the limit is 0, $f_2(n) = n$ grows slower than $f_4(n) = n^3$.

3. Compare $f_4(n) = n^3$ and $f_3(n) = e^n$:

   $$\lim_{n\to\infty} \frac{n^3}{e^n}.$$

   Using L'Hôpital's Rule:

   $$\lim_{n\to\infty} \frac{n^3}{e^n} = \lim_{n\to\infty} \frac{3n^2}{e^n} = \lim_{n\to\infty} \frac{6n}{e^n} = \lim_{n\to\infty} \frac{6}{e^n} = 0.$$

   Since the limit is 0, $f_4(n) = n^3$ grows slower than $f_3(n) = e^n$.

□

(b) [5 points] Prove or disprove: $(n^3 + n^2) \in O(n^3)$.

Solution:

Proof. $(n^3 + n^2) \in O(n^3)$.

1. We need constants $c > 0$ and $n_0 > 0$ such that:

   $$n^3 + n^2 \leq c \cdot n^3, \quad \forall n \geq n_0.$$

2. Dividing both sides by n^3:

   $$1 + \frac{1}{n} \leq c.$$

3. As $n \to \infty$, $\frac{1}{n} \to 0$.

4. One choice for n_0 and c is $n_0 = c = 2$.

$$1 + \frac{1}{2} = 1.5 \leq 2$$

5. $\frac{1}{n}$ decreases as n increases.

□

3. Recurrence Relations and Lower Bounds

(a) [5 points] Solve the following recurrence relation using the unfolding method:

$$T(n) = 4T\left(\frac{n}{2}\right) + n.$$

Show all steps and derive a closed-form solution.

Solution:

Proof. $T(n) = \Theta(n \log n)$.

$$\begin{aligned} T(n) &= 4T\left(\frac{n}{2}\right) + n \\ &= 4\left(4T\left(\frac{n}{4}\right) + \frac{n}{2}\right) + n = 16T\left(\frac{n}{4}\right) + n(1+2) \\ &= \dots \\ &= 4^k T\left(\frac{n}{2^k}\right) + n\sum_{i=0}^{k-1} 2^i = 2^{2k} T\left(\frac{n}{2^k}\right) + n(2^k - 1) \end{aligned}$$

Base case: $T(1) = \Theta(1) = c$. When $\frac{n}{2^k} = 1 \implies k = \log_2 n$:

$$T(n) = cn^2 + n^2 - n = \Theta(n^2)$$

□

(b) [5 points] Explain why the height of the decision tree for any comparison-based sorting algorithm must be at least $\lceil \log_2(n!) \rceil$.

Solution:

Proof.

1. The height of the decision tree of a comparison-based algorithm to sort n numbers is at least that of the decision tree of the optimal algorithm.
2. The decision tree of the optimal algorithm is a binary tree that and has to decide among $n!$ permutations. So it has $n!$ leaves.
3. Let the height of this decision tree be h. Then it has at most 2^h leaves.

4. Then, $n! \leq 2^h \implies h \geq \log_2 n!$. So the smallest integer value of h is $\lceil \log_2 n! \rceil$.

□