CSE 531: Algorithm Analysis and Design Spring 2025

Homework 1

Instructor: Kelin Luo, Chen Xu **Deadline: Feb/11/2025**

Your Name: _____ Your Student ID: _____

Problems	1	2	3	4	Total
Max. Score	10	15	15	15	55
Your Score					

Requirement:

- Save and submit your HW 1 submission to Brightspace as a single **typed PDF file**. Name your file: Your Last Name Your First Name YourStudents ID Number Assignment Number. Example: **Doe_John_55552222_HW1**
- You should view your submission after you upload it to make sure that it is not corrupted or malformed. Submissions that are rotated, upside down, or that do not load will not receive credit. Illegible submissions may also lose credit depending on what can be read. You are responsible for making sure your submission went through successfully.
- To create a figure, you have the option to use software such as "Ipe Drawing Editor" or other similar tools for digital illustrations. Alternatively, you may choose to draw the figure by hand and then insert it into your document.
- The HW 1 deadline is 11 Feb, 11:59PM EST. Late submissions (within 24 hours with 25 % penalty or 48 hours with a 50% penalty) will be recorded according to the submission timestamp on UBlearns. Submissions will close 48 hours after the deadline.
- Only the most recent submission is kept on Brightspace.
- The solutions will be released when the Homework 1 grades are published.
- Note that your total points for HW1 will not exceed 55. Suppose you receive $P1$, $P2$, $P3$, $P4$ points for each of the questions. Your HW1 grade, as counted towards your final grade, will be calculated as $5 * \min\{P1 + P2 + P3 + P4, 55\}/55$.
- You do not need to submit the questions or follow the provided HW1 question format; please submit only your answers.

Problem 1 (10 points). For each pair of functions f and g in the following table, please indicate whether $f = O(g)$, $f = \Omega(g)$ and $f = \Theta(g)$ respectively. The answer can be either "yes" or "no".

$f(n)$	$g(n)$	O	Ω	Θ
$2n$	n^2			
n^2	$4n^2$			
n^3	$4n^2 + 5n$			
$\log_2 n$	n			
$n^2 - 3n^2$	$5n^3$			
$n^2 - 3n^3$	$n^2 + n \log_2 n$			
$(n+1)!$	$n! + 100n^3$			
$n^{2 \log_2 n}$	2^n			
$\log_3 n$	$\log_2(n^{20})$			
$(n + \log_2 n)^4$	$(n^2 + n \log_2 n)^2$			

Problem 2 (15 points). For each of the following growth functions, simplify each of the following equations $f(n)$ (your simplified result should be a sum-free and logarithm-free equation in terms of n) and find specific values for constants c and n_0 to prove the requested bound. Show all work for you proof. **Answers given without valid work will receive no credit.** Hint: You may use any of the rules in the Reference Material and show your work as a sequence of steps.

(a) **(5 points)** Let $f(n) = 10n \log_2(2^n) + 0.2n$ and $g(n) = n^2$, prove $f(n) \in \Theta(g(n))$.

(b) **(5 points)** Let $f(n) = \sum_{i=1}^{n}(1 + \sum_{j=i}^{n} n)$ and $g(n) = n^3$, prove $f(n) \in \Theta(g(n))$.

(c) **(5 points)** Let $f(n) = 2^{2n+10\sqrt{n}+4}$ and $g(n) = 5^n$. Please indicate (answer: True or False) for the following questions and prove the bound if the answer is True.

- **(2 points)** whether $f(n) = O(g(n))$
- **(2 points)** whether $f(n) = \Omega(g(n))$
- **(1 points)** whether $f(n) = \Theta(g(n))$

Problem 3 (15 points).

(a) **(9 points)** Given an array of n non-negative integers, our goal is to check if there are two indices i and j in $[n]$ such that $A[j] = (A[i])^2$. Note that the two indices i and j can be the same. Consider the following three algorithms below and please briefly argue its correctness in no more than 3 sentences, give its (worst-case) running time and justify your answer in no more than 3 sentences. Your answers should be tight. (E.g, if an algorithm runs in $O(n)$ time, then the answer $O(n^2)$ is wrong even though $O(n) \subseteq O(n^2)$.)

(b) **(6 points)** Now suppose we have the same problem as Problem 3(a) except that the array A is sorted in increasing order. For each of the two algorithms Algorithm 2 and Algorithm 3, please briefly argue its correctness in no more than 3 sentences, give

its (worst-case) running time and justify your answer in no more than 3 sentences. Your answers should be tight.

Algorithm 1

```
1: for i ← 1 to n do
2:     for j ← 1 to n do
3:         if A[j] = (A[i])^2 then
4:             return "yes"
5: return "no"
```

Algorithm 2

```
1: j ← 1
2: for i ← 1 to n do
3:     while j ≤ n and A[j] < (A[i])^2 do
4:         j ← j + 1
5:     if j ≤ n and A[j] = (A[i])^2 then
6:         return "yes"
7: return "no"
```

Algorithm 3

```
1: for i ← 1 to n do
2:     ℓ ← i, r ← n
3:     while ℓ ≤ r do
4:         j ← ⌈(ℓ+r)/2⌉
5:         if A[j] = (A[i])^2 then
6:             return "yes"
7:         else if (A[i])^2 < A[j] then
8:             r ← j − 1
9:         else
10:            ℓ ← j + 1
11: return "no"
```

Hint: if you believe the algorithm is correct, please provide arguments for all possible inputs rather than relying on a single specific instance. To prove the correctness you need to show that: (i) the algorithm always terminates (i.e., it won't loop forever); (ii) when there do exists some pair i and j in $[n]$ such that $A[j] = (A[i])^2$ (note there can be multiple such pairs), the algorithm will always return yes; (iii) when no such pair exists, the algorithm can only return no. However, if you believe the algorithm is incorrect, you may use a specific instance to explain your argument. Moreover, if you believe the algorithm is incorrect, you do not need to analyze its running time.

Problem 4 (15 points). For problem (4a), you can either write down the edges or draw the DFS/BFS tree. For problem (4b), write your algorithm as pseudo code (in fewer than 15 lines), explain its correctness in no more than 3 sentences, and describe the runtime in no more than 3 sentences.

(a) **(6 points)** Consider the following undirected graph G in Figure 1 with non-negative edge weights. Please solve the following two problems with different starting vertices. Here we assume the vertices are explored in **lexicographic order**: for example, when you checking the neighbors of vertex s, $\{a, c, e\}$, you should first look at a, then c, then e.

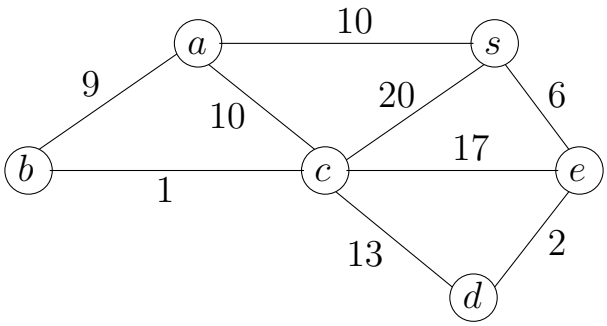

Figure 1: Traverse the graph using BFS and DFS

- **(3 points)** Using BFS to traverse the given graph starting from vertex s. List the edges included in the BFS algorithm.
- **(3 points)** Using DFS to traverse your graph starting from vertex a. List the edges included in the DFS algorithm.

(b) **(9 points)** A cycle in an *undirected graph* $G = (V, E)$ is a sequence of $t \geq 3$ different vertices $v_1, v_2, ..., v_t$ such that $(v_i, v_{i+1}) \in E$ for every $i = 1, 2, ..., t-1$ and $(v_t, v_1) \in E$. An **odd cycle** is defined as a cycle with an odd number of vertices. Given the adjacency-list representation of an undirected graph $G = (V, E)$, please design an $O(n + m)$-time algorithm to decide if G contains an **odd cycle** in G. Note that $n = |V|$ and $m = |E|$. Provide the Pseudo-code of your algorithm, and briefly argue the correctness of your algorithm and explain the running time of your algorithm. Hints: modify DFS or BFS or bipartiteness test algorithm.

Reference Material:

- Closed form summation equivalences

  The following works for any functions f, g (even constants). c is any constant relative to $i, j, k, \ell \in \mathbb{Z}$. Any sum $\sum_{i=j}^{k} f(i)$ is always 0 if $k < j$.

  S1. $\sum_{i=j}^{k} c = (k - j + 1)c$

  S2. $\sum_{i=j}^{k} (cf(i)) = c \sum_{i=j}^{k} f(i)$

  S3. $\sum_{i=j}^{k} (f(i) + g(i)) = \left(\sum_{i=j}^{k} f(i)\right) + \left(\sum_{i=j}^{k} g(i)\right)$

S4. $\sum_{i=j}^{k}(f(i)) = \left(\sum_{i=\ell}^{k}(f(i))\right) - \left(\sum_{i=\ell}^{j-1}(f(i))\right)$ ‘ (for any ‘$\ell < j$)

S5. $\sum_{i=j}^{k} f(i) = f(j) + f(j+1) + \ldots + f(k-1) + f(k)$

S6. $\sum_{i=j}^{k} f(i) = f(j) + \ldots + f(\ell-1) + \left(\sum_{i=\ell}^{k} f(i)\right)$ (for any $j < \ell \leq k$)

S7. $\sum_{i=j}^{k} f(i) = \left(\sum_{i=j}^{\ell} f(i)\right) + f(\ell+1) + \ldots + f(k)$ (for any $j \leq \ell < k$)

S8. $\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$

S9. $\sum_{i=0}^{k} 2^i = 2^{k+1} - 1$

- Closed form logarithm equivalences

  L1. $\log(n^a) = a\log(n)$

  L2. $\log(an) = \log(a) + \log(n)$

  L3. $\log(\frac{n}{a}) = \log(n) - \log(a)$

  L4. $\log_b(n) = \frac{\log_c(n)}{\log_c(b)}$

  L5. $\log(2^n) = 2^{\log(n)} = n$

- Example (for Problem 2 equations simplify): The derivation to find the closed form for $\sum_{i=0}^{n-2}\sum_{j=0}^{i} 20$ is as follows:
  apply S1 with $j = 0, k = i, c = 20$

$$\sum_{i=0}^{n-2}\sum_{j=0}^{i} 20 = \sum_{i=0}^{n-2}(i - 0 + 1)20 = \sum_{i=0}^{n-2}(i+1)20$$

  apply S6 with $j = 0, \ell = 1, k = n-2$

$$= 1 \cdot 20 + \sum_{i=1}^{n-2}(i+1)20 = 20 + \sum_{i=1}^{n-2}(i+1)20$$

  apply S2 with $c = 20, f(i) = (i+1), j = 1, k = n-2$

$$= 20 + 20\sum_{i=1}^{n-2}(i+1)$$

  apply S3 with $f(i) = i, g(i) = 1, j = 1, k = n-2$

$$= 20 + 20\left(\sum_{i=1}^{n-2} i + \sum_{i=1}^{n-2} 1\right)$$

  apply S8 with $k = n-2$

$$= 20 + 20\left(\frac{(n-2)(n-1)}{2} + \sum_{i=1}^{n-2} 1\right)$$

apply S1 with $c = 1, j = 1, k = n - 2$

$$= 20 + 20\left(\frac{(n-2)(n-1)}{2} + (n-2-1+1)\cdot 1\right)$$

$$= 20 + 20\left(\frac{n^2-3n+2}{2} + (n-2)\right)$$

$$= 20 + (10n^2 - 30n + 20) + (20n - 40)$$

$$= 10n^2 - 10n$$

- Example (for Problem 2 Asymptotic analysis): $f(n) = \log_2 n$, $g(n) = \log_8 n$
  ($f(n) = O(g(n))$ True) There exists a constant $c_1 = 4$ and $n_0 = 8$, for any number $n \geq n_0$, we have $f(n) = \log_2 n < 4\log_8 n$ because of $f(n) = \log_2 n = \frac{\log_8 n}{\log_8 2} = 3\log_8 n < 4\log_8 n$, implies that $f(n) = \log_2 n \leq c_1 * g(n)$, thus $f(n)$ is in $O(g(n))$.
  ($f(n) = \Omega(g(n))$ True) There exists a constant $c_1 = 1$ and $n_0 = 8$, for any number $n \geq n_0$, we have $f(n) = \log_2 n > \log_8 n = c_1 * g(n)$, thus $f(n)$ is in $\Omega(g(n))$.
  ($f(n) = \Theta(g(n))$ True) $f(n)$ is in $\Theta(g(n))$ since $f(n)$ is in $O(g(n))$ and $f(n)$ is in $\Omega(g(n))$.