國立成功大學

115學年度碩士班招生考試試題

編　號：179

系　所：數據科學研究所

科　目：計算機概論

日　期：0203

節　次：第 2 節

注　意：1.不可使用計算機
2.請於答案卷(卡)作答，於試題上作答，不予計分。

1. (36%) In the following statements, please specify if the statement is **True** or **False**. If the statement is True, explain why it is True. If it is False, give correct answer or explain why. (each 2%)
   (a) If an algorithm runs in $3n + 100$ steps, its time complexity is $O(n)$.
   (b) If algorithm A is $O(n \log n)$ and algorithm B is $O(n^2)$, then A is faster for every input size n.
   (c) Deleting a node from a singly linked list is always $O(1)$ if you are given only a pointer to the node to delete.
   (d) A binary heap provides a partial order, not a fully sorted structure.
   (e) An in-order traversal of a binary heap returns elements in sorted order.
   (f) BFS on an unweighted graph finds shortest paths measured by number of edges.
   (g) A hash table with separate chaining can have $O(n)$ worst-case lookup time.
   (h) A page fault always means the program is incorrect and must be terminated.
   (i) Round-robin scheduling always minimizes average waiting time.
   (j) In k-means clustering, each cluster center must be one of the original data points.
   (k) Adding more features always improves performance on unseen data in supervised learning.
   (l) Increasing a decision tree's maximum depth always improves test accuracy.
   (m) On highly imbalanced datasets, high accuracy can still mean poor performance on minority class.
   (n) A context switch typically saves and restores CPU registers and other execution state.
   (o) Virtual memory guarantees that a program will never run out of memory.
   (p) PCA can reduce dimensionality even when features are highly correlated.
   (q) Cross-entropy loss is only defined for binary classification problems.
   (r) In gradient descent, increasing the learning rate always speeds up convergence.

2. You are given a **large, undirected graph** modeling a city's transportation network. Each edge has a **"reliability score"** $r \in [0,1]$ (higher = more reliable). You want to find routes that are **most reliable overall.** To turn this into an optimization problem, define the **edge cost** as: $c(u,v) = -\log(r(u,v))$. So reliable edges give small cost; unreliable edges give large cost.
   (a) Propose a good data structure to store the graph when it is sparse. How would you store it if it becomes dense? Compare space complexity and explain when each is appropriate. (3%)
   (b) Which shortest-path algorithm would you use with the cost $c(u,v) = -\log r(u,v)$? State time complexity and explain why a naive approach like BFS is not suitable. (3%)
   (c) Suppose you are allowed to use **exactly one coupon** during your trip that can be applied to **one edge** on your path and makes that edge's cost become **0** (free), regardless of its original weight. You still want the **minimum total path cost** from source s to target t. Propose a simple algorithm idea to compute the best cost from s to t under this "use one coupon once" rule, and give the time complexity. (3%)

3. A data pipeline writes results to a log file. After a power loss, some "success" lines are missing even though the program printed "Saved!".
   (a) Explain the difference between **writing to a user-space buffer**, the **OS page cache**, and the **disk**. Why can data "disappear" after a crash? (3%)
   (b) Give two simple ways to make the log more crash-resilient, and the trade-off. (3%)

4. A shared server runs many training jobs. Some are short (2 minutes), some are long (2 hours). Users complain the machine "feels unfair."
   (a) Compare **FCFS (First-Come First-Served)** vs **Shortest Job First (SJF)** in terms of average waiting time and fairness. (3%)
   (b) Give a practical scheduling idea that balances responsiveness and fairness for mixed workloads. (3%)

5. A data science pipeline has two shared resources:
   - **Lock A**: protects access to a shared **feature store**
   - **Lock B**: protects access to a shared **model cache**

   Two worker programs run concurrently:
   - **Worker 1** does: acquire **Lock A**, then acquire **Lock B**, then release both.
   - **Worker 2** does: acquire **Lock B**, then acquire **Lock A**, then release both.

   Sometimes, the system freezes forever.
   (a) What is the most likely cause of the freeze? How can you explain it by a simple wait-for story? (3%)
   (b) Give one very practical fix that prevents this freeze, and briefly explain why it works. (3%)

6. Describe the differences between the following pairs of terms. (each 2%)
   (a) Precision vs. Recall
   (b) Process vs. Thread
   (c) Overfitting vs. Underfitting
   (d) Preemptive Scheduling vs. Non-preemptive Scheduling

7. You are given the following **undirected, unweighted** graph with vertices $\{A, B, C, D, E, F, G, H\}$, and edges: $(A, B), (A, C), (B, D), (B, E), (C, F), (E, F), (D, G), (F, H), (G, H)$.
   (a) Run **BFS starting from** A and list the vertices in discovery order (assume neighbors are explored in alphabetical order). Also write the distance (number of edges) from A to every vertex. (3%)
   (b) A vertex is called **critical** (for reaching H from A) if **removing that vertex** (and its incident edges) makes H **unreachable from** A. Is there any critical vertex other than A or H? If yes, give one. If no, say "none" and explain briefly. (3%)
   (c) You are allowed to add **exactly one new edge** anywhere between two currently non-adjacent vertices (a "teleport"). Your goal is to make the shortest distance from A to H become 2. Is it possible? If yes, give one edge to add. If no, explain why. (3%)

8. Answer the following questions on data science. (each 2%)
   (a) Explain **class imbalance** and give two simple ways to handle it. Why can accuracy be misleading here?
   (b) What is **cross-validation**? Why is it useful when data is limited?
   (c) Compare **training error** vs. **test error**. What does each indicate?
   (d) Compare **pretraining** vs. **fine-tuning**. Give a simple example.

9. A museum uses a simple **k-NN classifier** to decide whether a visitor is likely to buy a souvenir (**Buy** = +) or not (**No** = −) based on two features:
   - x_1: minutes spent in the gift shop area
   - x_2: number of items picked up and examined

   Training set (6 labeled points):

Points	x1	x2	Label
A	1.0	1.0	−
B	2.0	1.0	−
C	2.0	2.0	−
D	4.0	4.0	+
E	5.0	4.0	+
F	4.0	5.0	+

   Query point $Q = (3.0,3.0)$. Use **Euclidean distance**.
   (a) If $k = 3$, what label will k-NN predict for Q? Show the 3 nearest neighbors. (3%)
   (b) Assume point **C** was mislabeled and should actually be + (but the dataset still shows it as −). Which choice is *more robust* to this single mislabeled point for predicting Q: $k = 1$ or $k = 5$? Explain briefly. (3%)
   (c) Is k-NN with Euclidean distance always unaffected by feature scaling? Explain. (3%)
   (d) Give two simple 2-dimensional data situations where k-NN classification can perform poorly, and explain why. (3%)