

CS161 Final Exam

March 23, 2023

Instructions:

- This exam's duration is **180 minutes**.
- You may use two two-sided sheet of notes that you have prepared yourself. **You may not use any other notes, books, or online resources. You may not collaborate with others.**
- This exam has **5** problems over **17** pages, worth a total of **65** points. Please check that you have all of them both before beginning and before submission.
- **If you have a clarification question about the exam:** There are TAs outside the exam room to answer questions.
- There are extra pages at the end of the exam for scratch work. You may use it for additional space on problems if you need, but please indicate in the original space for the relevant problem if you have work on the back pages that you want to be graded, and label your work clearly.
- Please do not discuss exam questions or answers with other students until solutions are posted.
- You may cite any result we have seen in lecture without proof unless otherwise stated.
- **Please write your name at the top of all pages.**

Advice: If you get stuck on a problem, move on to the next one. Pay attention to how many points each problem is worth. Read the problems carefully.

The following is a statement of the Stanford University Honor Code:

1. *The Honor Code is an undertaking of the students, individually and collectively:*
 - (1) *that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;*
 - (2) *that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.*
2. *The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.*
3. *While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.*

By signing your name below, you acknowledge that you have abided by the Stanford Honor Code while taking this exam.

Signature: _____

Name: _____

SUNetID: _____

1 True or False (6 pt.)

For each of the following questions, select whether the given statement is True or False. **No explanation is required for True/False questions.** Please clearly mark your answers; if you must change an answer, either erase it thoroughly or very clearly indicate which answer you have chosen. **Ambiguous answers will be marked incorrect.**

- 1.1. **(1 pt.)** Floyd-Warshall is always asymptotically slower than Bellman-Ford (for finding shortest paths between all pairs of nodes).

True False

- 1.2. **(1 pt.)** Kruskal's MST algorithm is an example of a greedy algorithm that grows a forest.

True False

- 1.3. **(1 pt.)** You can detect whether a graph $G = (V, E)$ is bipartite in $O(|V| + |E|)$ time.

True False

- 1.4. **(1 pt.)** Any algorithm that can be solved optimally using Dynamic Programming can also be solved using a Greedy algorithm

True False

- 1.5. **(1 pt.)** The SCCs returned by Kosaraju's algorithm can be different based on which node the algorithm starts at.

True False

- 1.6. **(1 pt.)** Amortized runtime of insertion is $O(\log(n))$ for all binary search trees.

True False

2 Multiple Choice (10 pt.)

For each of the following questions, select **all correct answer choices**. **No explanation is required.** Please clearly mark your answers; if you must change an answer, either erase it thoroughly or very clearly indicate which answer you have chosen. **Ambiguous answers will be marked incorrect.**

- 2.1. **(2 pt.)** Suppose that H is a hash family of size s , and we choose h randomly from H . Which of the following options imply by themselves that an item can be found using h in constant time? Let n be the number of buckets h hashes to, and let u be the size of the universe of keys.

- A) $\mathbb{P}(h(u_i) = h(u_j)) \leq \frac{1}{n^2}$ when $u_i \neq u_j$
- B) $\mathbb{P}(h(u_i) = h(u_j)) \leq 0.0001$ when $u_i = u_j$
- C) For every bucket b , $\mathbb{E}[\text{number of items in } b] \leq 2$
- D) For any item u_i , $\mathbb{E}[\text{number of items in } u_i\text{'s bucket}] \leq 2$

- 2.2. **(2 pt.)** Which of the following algorithms can be used to find shortest paths on all graphs with negative edges?

- A) Dijkstra
- B) Floyd-Warshall
- C) BFS
- D) DFS
- E) Bellman-Ford

- 2.3. **(2 pt.)** Recall the 0/1 Knapsack problem from lecture, where the objective is to maximise the value of the knapsack given that we have a single copy of each item. Select the options that are *false*.

- A) The optimal solution cannot always be obtained using a greedy algorithm.
- B) A smaller capacity Knapsack is sufficient optimal sub-structure to solve this problem if using DP.
- C) There is a DP solution for this algorithm that can be easily modified to return the contents of the knapsack along with the value of the knapsack.
- D) Bottom-up algorithms are usually recursive whereas top-down algorithms are usually iterative.

- 2.4. **(2 pt.)** The shortest path from a source node to all other nodes in a graph will remain unchanged if:
- A) A positive constant value is added to all edge weights.
 - B) A positive constant value is multiplied by all edge weights.
 - C) A positive constant value is subtracted from all edge weights.
 - D) All edge weights are divided by a positive constant value.
- 2.5. **(2 pt.)** Which of the following are not characteristics of wicked problems? Select all that apply.
- A) Their solutions are either correct or incorrect.
 - B) They have no definitive and exhaustive formulation.
 - C) They are symptomatic of, and interconnected with, other problems.
 - D) Solutions may be tested as many times as is necessary.
 - E) None of the above.

3 Short Answers (16 pt.)

3.1 Central Location (3 pt.)

Suppose you are given a connected directed weighted graph with vertices V . Devise an algorithm to find the node u in V which minimizes the quantity $\sum_{v \in V, u \neq v} \text{dist}(u, v)$, where $\text{dist}(u, v)$ returns the length of the shortest path between u and v . Your algorithm should run in time $O(n^3)$.

[We are expecting: Pseudocode OR an English description of your algorithm, along with a proof of runtime.]

3.2 Longest Path (3 pt.)

Suppose you are given a weighted, directed, acyclic graph. Devise an algorithm to find the longest path between two vertices a and b .

[We are expecting: Pseudocode OR an English description of your algorithm. No proof of runtime is necessary, but you should pick an algorithm that is relatively efficient.]

3.3 Species Extinction (3 pt.)

Suppose you are given a list of n species and a list of m statements of the form “if species x goes extinct, then species y will also go extinct.” Construct an algorithm to find the maximum set S of species such that if any one species in S goes extinct, then all the rest of the species in S will eventually go extinct. Your algorithm should run in time $O(m + n)$.

For example, if your species are [Rhinos, Hippos, Hawks], and your extinction relationships are [(Hawks extinct \rightarrow Rhinos extinct), (Rhinos extinct \rightarrow Hippos extinct), (Hippos extinct \rightarrow Hawks extinct)], then you should return the set $S = [\text{Rhinos, Hippos, Hawks}]$.

[We are expecting: Pseudocode OR an English description of your algorithm, along with a proof of runtime.]

3.4 Baby Hashing (3 pt.)

Suppose that you are given access to an unlimited number of independent random hash functions that each output 5 digit numbers such that for any two distinct elements x, y in your domain and each randomly sampled hash function h , $p(h(x) = h(y)) \leq 1/10$. One call of $h(x)$ runs in time $O(1)$. Construct a new random hash function h that outputs upto $O(n)$ -digit numbers such that $p(h(x) = h(y)) \leq 2^{-n}$, and such that $h(x)$ can be computed in time $O(n)$.

[We are expecting: A definition of your hash function, along with a brief explanation for why it works.]

3.5 DP runtime (4 pt.)

Consider the DP algorithm below:

```
global array A
# initialize A as an array of length n+1, all values None
def fooDP(x):
    if x < 1:
        return 0
    if A[x] != None:
        return A[x]
    y = x
    for i in range(log(x)):
        y = hash(y)
    y = y + fooDP(x/2) + fooDP(x/4)
    A[x] = y
    return y
```

Assume that n is a power of 2. What is the runtime of $\text{fooDP}(n)$? Assume **hash** runs in constant time.

[We are expecting: The runtime, along with a thorough proof.]

4 Graph Algorithm Analysis (23 pt.)

Let $G = (V, E)$ be a directed, weighted graph with the following properties:

- V is a set of n vertices.
- E is a set of m edges.
- For edge $(x, y) \in E$, let $w(x, y)$ be its *possibly negative* edge weight.
- Let $p = [x_1, \dots, x_i]$ be a path of vertices in V . We define the path length $l(p) = \sum_{j=1}^{i-1} w(x_j, x_{j+1})$. In other words, the path length is the sum of all edge weights along the path.
- For $x \in V$ and $z \in V$, let distance $d(x, z)$ be the shortest path length from x to z .

4.1 (4 pt.)

For $x \in V$, let $h(x) = \min_{z \in V} (d(z, x))$. In other words, $h(x)$ is the shortest distance from *any* vertex to x (including from x to itself).

Describe an algorithm that computes $h(x)$ for all $x \in V$ in $O(mn)$ time if G has no negative cycles, and returns False otherwise.

[We are expecting: *Clear English Description OR Pseudocode, along with a proof of run-time and correctness***]**

[Hint: *Try adding an extra node to the graph***]**

4.2 (3 pt.)

Let p be a path in G with starting vertex a and ending vertex b . Prove that $l(p) + h(a) \geq h(b)$, assuming G has no negative cycles.

[We are expecting: A thorough proof of the statement]

4.3 (4 pt.)

Let G' be a directed, weighted graph with the following construction:

- $G' = (V, E)$. In other words, G' has the same set of vertices and edges as G .
- Let $w'(x, y)$ be the weight of edge (x, y) in G' . We define $w'(x, y) = w(x, y) + h(x) - h(y)$.

For a path $p = [x_1, \dots, x_i]$ with nonzero length, let $l'(p)$ be the length of p in G' . Prove that $l'(p) = l(p) + h(x_1) - h(x_i)$

[We are expecting: A thorough proof of the statement]

4.4 (3 pt.)

Conclude that all edge weights in G' are non-negative.

[**We are expecting:** *A thorough proof of the statement*]

4.5 (4 pt.)

Let there be a path from a to b in G and G' . Show that if p is a shortest path from a to b in G , then it is also a shortest path from a to b in G' .

[**We are expecting:** *A thorough proof of the statement*]

4.6 (5 pt.)

Describe an algorithm that computes the all-pairs-shortest-paths problem for G in $O(mn + n^2 \log n)$ time if G has no negative cycles, and returns False otherwise. Feel free to use results from the previous parts.

[We are expecting: *Clear English description OR pseudocode, along with a proof of run-time and correctness***]**

5 Table Making (10 pt.)

You are the manager of a small factory that produces handcrafted furniture. Your factory has been growing in popularity, and you've decided to hire a team of n skilled workers to help you meet the increasing demand for your products.

One day, you receive a big order for a set of m custom-designed dining tables, each of which requires a specific set of skills to complete. You need to decide which workers to assign to each task while taking into account their availability and skill sets.

Suppose each worker i has his/her own flexible schedule, and can only work on certain days $D_i \subseteq [1, N]$. Also, since different tables require different skills to make, not all workers can make all the tables. For every table j , we use $W_j \subseteq [1, n]$ to denote the set of workers that can make it. It always takes exactly one day for a worker to make a customized table.

Assume that $m = \Theta(n)$, $N = \Theta(n)$, $\sum_{j=1}^m |W_j| = \Theta(n)$, $\sum_{i=1}^N |D_i| = \Theta(n^2)$ for the purposes of runtime calculation.

5.1 (5 pt.)

Design a max-flow algorithm that can help you decide what is the max number of custom tables that can be made by these workers. The runtime of your algorithm should be $O(n^2)$.

[We are expecting: *Clear English description OR pseudocode, along with a proof of runtime and correctness***]**

5.2 (5 pt.)

It turns out that your factory has limited working space, and it can sometimes become too crowded if too many workers show up. As a result, you've recently decided that at most k workers can come to work on any given day. Modify your algorithm to compute the max number of tables that can be made after you enforce this policy. The runtime of your algorithm should be $O(n^3)$.

[We are expecting: *Clear English description OR pseudocode, along with a proof of run-time and correctness***]**

This page is intentionally blank for extra space for any question.

Please indicate in the relevant problem if you have work here that you want to be graded, and label your work clearly.

This page is intentionally blank for extra space for any question.

Please indicate in the relevant problem if you have work here that you want to be graded, and label your work clearly.

This page is intentionally blank for extra space for any question.

Please indicate in the relevant problem if you have work here that you want to be graded, and label your work clearly.

This page is intentionally blank for extra space for any question.

Please indicate in the relevant problem if you have work here that you want to be graded, and label your work clearly.