U.C. Berkeley — CS170 : Algorithms
Lecturers: Alessandro Chiesa and Satish Rao

Midterm 2
October 23, 2018

Midterm 2

Name:

SID:

Name and SID of student to your left:

Name and SID of student to your right:

Exam Room:

☐ 2040 VLSB ☐ 2050 VLSB ☐ 2060 VLSB ☐ Hearst Field Annex A1
☐ 160 Kroeber ☐ 100 GPB ☐ 10 Evans ☐ 60 Evans ☐ HP Auditorium
☐ 145 Dwinelle ☐ 310 Soda (6-8 pm) ☐ 310 Soda (8-10 pm) ☐ Other

Please color the checkbox completely. Do not just tick or cross the box.

Rules and Guidelines

- **The exam is out of 144 points and will last 110 minutes.**
- Answer all questions. Read them carefully first. Not all parts of a problem are weighted equally.
- Write your student ID number in the indicated area on each page.
- Be precise and concise.
- When there is a box for an answer, **write in the solution box provided.**
- You may use the blank page on the back for scratch work, but it will not be graded. Box numerical final answers.
- Question 1 is true/false. Questions 2-6 are short answer, roughly sorted by topic. Questions 7-9 are algorithm design problems.
- **The problems do not necessarily follow the order of increasing difficulty.** ***Avoid getting stuck on a problem.***
- Any algorithm covered in lecture can be used as a blackbox. Algorithms from homework need to be accompanied by a proof or justification as specified in the problem.
- You may assume that comparison of integers or real numbers, and addition, subtraction, multiplication and division of integers or real or complex numbers, require $O(1)$ time.
- **Warmup Beginning Italian Question:** What does the "ch" in bruschetta sound like? **(0 points)**
  ○ 'kuh' as in cat. ○ 'chuh' as in cheese.
- Good luck!

Discussion Section

Which of these do you consider to be your primary discussion section(s)? Feel free to choose multiple, or to select the last option if you do not attend a section. **Please color the checkbox completely. Do not just tick or cross the boxes.**

- ☐ Perla, Monday 9 - 10 am, Dwinelle 243
- ☐ Jenny, Monday 9 - 10 am, Soda 320
- ☐ Sean, Monday 10 - 11 am, Cory 241
- ☐ Yining, Monday 10 - 11 am, Wheeler 222
- ☐ Jerry, Monday 11 am - 12 pm, Etcheverry 3113
- ☐ Jeff, Monday 11 am - 12 pm, Dwinelle 130
- ☐ Peter, Monday 12 - 1 pm, Evans 3
- ☐ David, Monday 12 - 1 pm, Wheeler 108
- ☐ Nate, Monday 12 - 1 pm, Soda 320
- ☐ James, Monday 1 - 2 pm, Dwinelle 182
- ☐ Mudit, Monday 1 - 2 pm, Soda 320
- ☐ Arun, Monday 1 - 2 pm, Barker 110
- ☐ Jierui, Monday 2 - 3 pm, Evans 9
- ☐ Simin, Monday 2 - 3 pm, Evans 70
- ☐ Brandon, Monday 2 - 3 pm, Dwinelle 242
- ☐ Ming, Monday 3 - 4 pm, Cory 289
- ☐ Harley, Monday 3 - 4 pm, Evans 9
- ☐ Aarash, Monday 4 - 5 pm, Dwinelle 79
- ☐ Vinay, Monday 4 - 5 pm, Etcheverry 3119
- ☐ Zheng, Monday 4 - 5 pm, Evans 70
- ☐ Zihao, Monday 5 - 6 pm, Dwinelle 79
- ☐ Max, Monday 5 - 6 pm, Dwinelle 243
- ☐ Matthew, Tuesday 9 - 10 am, Wheeler 108
- ☐ Ajay, Tuesday 9 am - 10 am, Etcheverry 3113
- ☐ Nick, Tuesday 11 am - 12 pm, Etcheverry 3111
- ☐ Sam, Tuesday 12 - 1 pm, Etcheverry 3111
- ☐ Julia, Tuesday 1 - 2 pm, Etcheverry 3119
- ☐ Don't attend Section.

1 True/False

3 points each.

(a) The MST of G is unique if and only if G has distinct edge weights.

○ True ○ False

(b) The spanning tree of maximum weight in G is the minimum spanning tree in a copy of G with all edge weights negated.

○ True ○ False

(c) Let G be a weighted undirected graph with positive edge weights where edge e has weight w_e for all $e \in E$, and G' be a copy G except that every edge e has weight w_e^2. Any MST of G' is an MST of G.

○ True ○ False

(d) In Huffman coding, assuming distinct frequencies, the character with the largest frequency has depth (distance to root) less than or equal to the depth of every other character in the Huffman tree.

○ True ○ False

(e) If a satisfying assignment for a Horn formula exists, the greedy algorithm for Horn formulas finds the satisfying assignment that sets the fewest variables to true among all satisfying assignments.

○ True ○ False

(f) If a linear program has more than one optimal solution, then it has infinitely many.

○ True ○ False

(g) Recall that in a zero sum game, a player has a best defense strategy: a strategy that will achieve the optimal expected payoff even if the opponent knows the strategy beforehand. It is always best to play this strategy, *even if you know ahead of time your opponent is playing some non-optimal strategy*.

○ True ○ False

(h) If we multiply all capacities by a constant c, then the max-flow of the graph will by multiplied by c.

○ True ○ False

(i) If all edge capacities in a max flow problem instance are distinct, the maximum flow is unique.

○ True ○ False

(j) Suppose the maximum (s, t)-flow of some graph has value f. Now we increase the capacity of every edge by 1. Then the maximum (s, t)-flow in this modified graph will have value at most $f + 1$.

○ True ○ False

2 Quick Fixes.

(a) Suppose you are given a MST T in a graph G. Consider further that a tree edge, $e \in T$, has its weight increased. Give a linear time method to find an MST in the graph with new weights. You must justify the correctness of your answer, but need not provide any runtime analysis. (**6 points**)

(b) Consider a knapsack instance with n items where item i has weight w_i and value v_i. Suppose you have a solution for the knapsack problem **with repetition** of weight W. Briefly describe as efficient a method as possible to compute a solution with the addition of an $(n+1)$th item with weight w_{n+1} and value v_{n+1}. State your running time. No justification is needed.

(You may assume that you still have the table where $K(w)$ for $w \in [0, W]$ corresponds to the highest value knapsack of weight w. To be clear, repetition is allowed.) (**6 points**)

3 A little bit of greed.

(a) Suppose you have an undirected graph $G = (V, E)$, with average degree d (or equivalently $|E| = d|V|/2$).

Assume d is even for simplicity. We want to find some $V' \subseteq V$, such that each $v \in V'$ has at least $d/2$ neighbors in V'.

Consider the algorithm that begins with $V' = V$, and repeatedly removes any vertex from V' which has strictly less than $d/2$ neighbors in V' until no such vertices exist in V'.

1 This algorithm terminates with a non-empty set V' where every vertex $v \in V'$ has at least $d/2$ neighbors in V'. (**2 points**)

○ True ○ False

2 If you answered **True**, provide a proof. If you answered **False**, give an example graph where the algorithm removes all vertices. (**6 points**)

(b) In homework, you saw the following problem: You have G, with weights $\ell(v) \geq 0$ and one proceeds by marking vertices until all vertices are marked. Furthermore, when marking a vertex u the player receives $\sum_{v \in M(u)} \ell(v)$ points where $M(u)$ is the set of marked neighbors of u.

1 The number of points for the strategy of marking the vertices in decreasing order of $\ell(v)$ is $\sum_{e=(u,v) \in E}$ _____. Fill in the blank (put your answer in the box). (**2 points**)

2 Give a one line argument based on the previous part that this order is optimal. (**2 points**)

(c) Recall the greedy set cover algorithm which repeatedly adds the set with the most uncovered elements to its solution until all elements are covered.

Suppose we run the greedy set cover algorithm on an instance where there are 9 elements in the union of all the sets, and the optimal solution uses 3 sets to cover them. What is the largest possible number of sets in the solution output by the greedy set cover algorithm? Note: You do not need to compute or estimate any logarithms to answer this question. (**3 points**)

4 Now for dynamism.

(a) In the **min weight** knapsack problem **without repetition**, one is given n items where item i has weight w_i and value v_i and one wishes to find the minimum weight set of items with value at least V. We define a subproblem $K(v, i)$ to be the minimum weight of any knapsack of value at least v using the first i items.

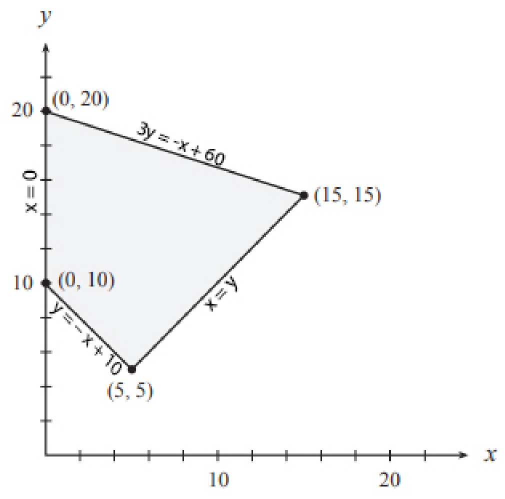
1 Give the recurrence for $K(v, i)$. (No justification needed) (**4 points**)

2 What is the **minimum space complexity** of solving the recurrence above? (**2 points**)

5 Linear Programming.

(a) What is the optimal value of the linear program max $x_1 + x_2$, where $x_1 + 2x_2 \leq 3$, and $x_1, x_2 \geq 0$? (**3 points**)

(b) Suppose that you plot the feasible region from a linear program and get the following graph (the shaded region is feasible). Assuming that your objective is min $x + 2y$, what is the value of the optimal solution? (**3 points**)



(c) Consider a linear program with n variables and m inequalities (including non-negativity constraints) where all variables are unrestricted.

1 A vertex of the linear program is a point where at least ____ inequalities hold with equality. (**2 points**)

2 For two points which are neighboring vertices at least ____ inequalities hold with equality at both points. (**2 points**)

(d) Express the constraint $|x| \leq 4$ as a set of linear inequalities. (Do not introduce new variables in your solution.) (**3 points**)

(e) We wish to find $x, y \geq 0$ that minimizes $|ax + by - 1|$. Write a linear program for this problem. (**4 points**)

For each of the LPs in parts (f), (g), (h), decide if it is infeasible, unbounded, has a unique solution, or has multiple solutions.

(f) Objective function: $\max x_1 + 2x_2$; constraints: $x_1 + x_2 \geq 10; x_1 \leq 0$ (**2 points**)

○ infeasible ○ unbounded ○ unique solution ○ multiple solutions

(g) Objective function: $\max x_1 + x_2$; constraints: $x_1 + x_2 \leq 3; x_1, x_2 \geq 0$ (**2 points**)

○ infeasible ○ unbounded ○ unique solution ○ multiple solutions

(h) Objective function: $\min 3x_1 - 2x_2$; constraints: $x_1 + x_2 \leq 10; 2x_1 + 2x_2 \geq 30$ (**2 points**)

○ infeasible ○ unbounded ○ unique solution ○ multiple solutions

(i) Write the dual of the following LP: (**4 points**)

$$\begin{aligned} \max\ & 2x_1 + 3x_2 \\ & x_1 + 4x_2 \leq 5 \\ & 3x_1 + 2x_2 \leq 4 \\ & x_1, x_2 \geq 0 \end{aligned}$$

(j) Consider the **zero sum game** represented by the matrix, where each entry in the matrix corresponds to the row player's payoff.

$$\begin{bmatrix} 4 & -3 \\ -2 & 1 \end{bmatrix}$$

(1) What is the expected payoff if the column player plays $(2/5, 3/5)$ and the row player picks the best response to this strategy? (Recall that the row player is trying to maximize the payoff.) (**2 points**)

(2) What is the expected payoff if the row player plays $(1/2, 1/2)$ and the column player plays the best response to this strategy? (**2 points**)

(3) Is the expected payoff positive, negative, or zero when both players play optimally? (**1 point**)

○ positive ○ negative ○ zero

6 In the flow.

For all parts of this problem, let G be a directed graph with source s and sink t, where **all edge capacities are** 1**.**

(a) (True/False). Consider running the Ford-Fulkerson algorithm to find a max s-t flow in G. If G is a DAG, then during the course of the algorithm, every edge capacity or residual reverse edge capacity in the residual graph will be 0 or 1. (**2 points**)

(Ford-Fulkerson is the name of the algorithm which repeatedly finds an s-t path in the residual graph and pushes as much flow as on possible on this path.)

○ True ○ False

Let $\delta(v, G) = \text{indegree}(v, G) - \text{outdegree}(v, G)$, where $\text{indegree}(v, G)$ is the total capacity of edges pointing to v in G, and $\text{outdegree}(v, G)$ is the total capacity of edges pointing out of v in G. **For the following 3 parts your answer should be in terms of some subset of $\delta(v, G)$, indegree(v, G), outdegree(v, G) (for any vertex v) and f.**

(b) If one has routed f units of flow from s to t and the resulting residual graph is G', what is $\delta(v, G')$ when $v \notin s, t$? (**3 points**)

(c) If one has routed f units of flow from s to t and the resulting residual graph is G', what is $\delta(s, G')$? (**3 points**)

(d) If one has routed f units of flow from s to t and the resulting residual graph is G', what is $\delta(t, G')$? (**3 points**)

7 Merging Piles

Given n sandpiles of weights $w_1, \ldots, w_n$ on a line, we want to merge all the sand piles together. At each step we can only merge **adjacent** sand piles, with cost equal to the weight of the merged sandpile.

In particular, merging sandpiles i and $i+1$ has cost $w_i + w_{i+1}$. Furthermore, one gets a single sandpile of weight $w_i + w_{i+1}$ in its place which is now adjacent to sandpiles $i-1$ and $i+2$.

Note that different merging orders will result in different final costs, and we wish to find the order that minimizes that cost.

(a) What's the minimum cost of merging the four piles $(100, 1, 1, 100)$? (**2 points**)

(b) You will give a dynamic programming algorithm for this problem. (**12 points**)

(1) Define your subproblems.

(2) Describe and justify the recurrence for computing the solution to a subproblem.

(3) Describe the base cases or base subproblems.

(4) What is the time and space complexity of computing your solution?

8 Setting up (Integer) Linear Programs.

Integer linear programming is very general in that we can formulate a lot of problems using it.

A matching in a graph $G = (V, E)$ is a subset of edges, $M \subseteq E$, where no vertex in V is incident to more than one edge in M.

Formulate the problem of finding the maximum sized matching for an undirected graph $G = (V, E)$ as an **integer linear program**.

In addition to linear constraints, you may include constraining values of variables to come from a specific set, e.g., the constraint $x \in \{0, 1, 2, 3, 4\}$ indicates that x is an integer in $[0, 4]$. For each part below, you must state what is asked **both mathematically and in words**. (Only the description of each part is needed, no justification required) (**8 points**)

(1) Variables:

(2) Objective:

(3) Constraints:

9 Interval Covers.

Given an ordered set of points $a_1, \ldots, a_n$, we wish to cover them with **exactly** K intervals. For example, the points $\{1,3,4,7\}$ are covered by the 2 intervals $\{[1,3],[4,7]\}$, or by intervals $\{[1,1],[3,7]\}$ (i.e., zero length intervals are fine.)

(a) The ℓ_1-cost of a set of intervals $\{[x_1,y_1],\ldots,[x_k,y_k]\}$ is $\sum_k |x_k - y_k|$. Give an efficient algorithm that given an ordered set of n points produces the minimum ℓ_1-cost K interval cover. Briefly justify correctness and state the time and space complexity. (**8 points**)

Algorithm:

Justification.

Time/Space complexity.

(b) The ℓ_2-cost of a set of intervals $\{[x_1,y_1],\ldots,[x_k,y_k]\}$ is $\sum_k |x_k - y_k|^2$.

Give an efficient algorithm that given an ordered set of n points produces the minimum ℓ_2-cost K interval cover. Briefly justify correctness and state the time and space complexity. (**8 points**)

Algorithm:

Justification.

Time/Space complexity.