PRINCETON UNIVERSITY FALL '14 COS 521: ADVANCED ALGORITHMS

Homework 2

Out: *Oct 7* Due: *Oct 16*

You can collaborate with your classmates, but be sure to list your collaborators with your answer. If you get help from a published source (book, paper etc.), cite that. The answer must be written by you and you should not be looking at any other source while writing it. Also, limit your answers to one page, preferably less —you just need to give enough detail to convince the grader.

Typeset your answer in latex (if you don't know latex, scan your handwritten work into pdf form before submitting). To make things easier to grade, submit answers in the numbered order listed below, and also make sure your name appears on every page.

§1 Draw the full tree of possibilities for the cake-eating problem discussed in class, and compute the optimum cake-eating schedule. To keep the tree size manageable, draw it with the following slight changes: The amount you eat each day has to be an integer multiple of 1/3, and on each day your roommates will with probability 1/2 eat 40% of the cake.) (If multiple branches or sub-branches are identical, you may label the branch with a variable and use the variable in lieu of re-drawing the branch. You may also omit branches where you eat 0 cake.)

§2 (*Stable Matchings with Real-Valued Utilities*) We saw stable matchings in a guest lecture. Another formulation of the bipartite stable matching problem has each agent i submit a real number $u_i(j)$ for each element j in the opposite partition, representing the utility of being matched with that element. We then define a perfect matching M to be *stable* if there does not exist a pair $(v, w) \notin M$ such that both $u_v(w) > u_v(v')$ and $u_w(v) > u_w(w')$ where both (v, v') and (w, w') are in M.

(a) Prove that if the two partitions of the graph are of equal size, $u_v(w) = u_w(v)$ for all pairs (v, w), and $u_v(w) \neq u_{v'}(w')$ for all $\{v, w\} \neq \{v', w'\}$ then there exists a unique stable matching among the agents.

(b) Show by example that if we remove the final condition (that utilities are unique between different pairs of agents) from part (a), then the instance can contain multiple stable matchings.

§3 In ℓ_2 regression you are given datapoints $x_1, x_2, \ldots, x_n \in \Re^k$ and some values $y_1, y_2, \ldots, y_n \in \Re$ and wish to find the "best"linear function that fits this dataset. A frequent choice for best fit is the one with *least squared error*, i.e. find $a \in \Re^k$ that minimizes

$$\sum_{i=1}^{h} |y - a \cdot x_i|^2.$$

Show how to solve this problem in polynomial time (hint: reduce to solving linear equations).

§4 (Firehouse location) Suppose we model a city as an m-point finite metric space with $d(x, y)$ denoting the distance between points x, y. The city has n houses located at points $v_1, v_2, \ldots, v_n$ in this metric space. The city wishes to build k firehouses and asks you to help find the best locations $c_1, c_2, \ldots, c_k$ for them, which can be located at any of the m points in the city. The *happiness* of a town resident with the final locations depends upon his distance from the closest firehouse. So you decide to minimize the cost function $\sum_{i=1}^{n} d(v_i, u_i)$ where $u_i \in \{c_1, c_2, \ldots, c_k\}$ is the firehouse closest to v_i. Describe an LP-based algorithm that runs in $poly(m)$ time and solves this problem approximately. If OPT is the optimum cost of a solution with k firehouses, your solution is allowed to use $O(k \log n)$ firehouses and have cost at most $(1 + \epsilon)$OPT.

§5 In class we designed a 3/4-approximation for MAX-2SAT using LP rounding. Extend it to a 3/4-approximation for MAX-SAT (i.e., where clauses can have 1 or more variables). Hint: you may also need the following idea: if a clause has size k and we randomly assign values to the variables (i.e., 0/1 with equal probability) then the probability we satisfy it is $1 - 1/2^k$.

§6 You are given data containing grades in different courses for 5 students. As discussed in Lecture 5, we are trying to "explain" the grades as a linear function of the student's aptitude, the easiness of the course and some error term. Denoting by Grade_{ij} the grade of student i in course j this linear model hypothesizes that

$$\text{Grade}_{ij} = \text{aptitude}_i + \text{easiness}_j + \epsilon_{ij},$$

where ϵ_{ij} is an error term.

As we saw in class, the problem of finding the best model that minimizes the sum of the $|\epsilon_{ij}|$'s can be solved by an LP. Your goal is to use any standard package for linear programming (Matlab/CVX, Freemat, Sci-Python, Excel etc.; we recommend CVX on matlab) to fit the best model to this data. Include a printout of your code, and the calculated easiness values of all the courses and the aptitudes of all the students.

	MAT	CHE	ANT	REL	POL	ECO	COS
Alex			C+	A	B+	A-	C+
Billy	B+	A-			A-	B	B
Chris	B	B+			A	A-	B+
David	A		B-	A		A-	
Elise		B-	C	B+	B	B	C

Assume $A = 10, B = 8$ and so on. Let $B+ = 9$ and $A- = 9.5$. (If you use a different numerical conversion please state it clearly.)

§7 (Optimal life partners via MDP) Your friend is trying to find a life partner by going on dates with n people selected for her by an online dating servie. After each date she has two choices: select the latest person she dated and stop the process, or reject this person and continue to date. She has asked you to suggest the optimum *stopping rule*. You can assume that the n persons are all linearly orderable (i.e. given a choice between any two, she is not indifferent and prefers one over the other). The dating service presents the n chosen people in a random order, and her goal is to maximise the chance of ending up with the person that she will like the most among these n. (Thus ending up even with her second favorite person out of the n counts as failure; she's a perfectionist.) Represent her actions as an MDP, compute the optimum strategy for her and the expected probability of success by following this strategy.

(Hint: The Optimal rule is of the form: *Date γn people and decide beforehand to pass on them. After that select the first person who is preferable to all people seen so far.* You may also need that $\sum_{k=t_1}^{t_2} \frac{1}{k} \approx \ln \frac{t_2}{t_1}$.)

§8 (extra credit) In question 4 try to design an algorithm that uses k firehouses but has cost $O(\mathrm{OPT})$. (Needs a complicated dependent rounding; you can also try other ideas.) Partial credit available for partial progress.