Discrete Optimization: Home Exam

Chalmers, Period 3, 2017 (TDA206/DIT370)
Instructor: John Wiedenhoeft
Examiners: Devdatt Dubhashi, Peter Damaschke

Instructions:

- There are 35 points in total for this exam. For the overall grading of this class, please refer to the course website.
- You have until March 16, 2017, 10:00 am to finish this exam and upload it to the FIRE system, just as you did for the homework. Typed submissions and handwritten scans are both fine. Submissions must be legible after printing on A4 paper. All submissions must be in PDF format.
- Please start each problem on a new page (if you submit a LaTeX solution, the command for that is \newpage). Subproblems may be on the same page. For instance, (1a) and (1c) may be on the same page, but (1a) and (2c) must be on different pages.
- Include a cover sheet containing your name as the first page (you may use the page you are reading right now). Do NOT write any solutions on the cover sheet, it will not be considered for grading. Do NOT write your name or other identifying information on any other page.
- All work must be your own. You MAY use whatever tools and sources are available to you. However, you may NOT invoke the help of others, be it your classmates or people on the internet. For example, you MAY use existing answers on StackExchange.com to help you solve the problems, but you may NOT post exam questions there and ask for help. It is your responsibility to ensure that sources are reliable and information found there is correct, so use external sources at your own risk (Exception: Potential errors in the lecture notes or the suggested literature will not be counted against you, of course). Please cite your sources!

Question:	1	2	3	4	5	Total
Points:	6	4	6	13	6	35
Score:						

Name: ______________________________

*This page is intentionally left blank.
Do not write on this page, it will
not be considered for grading!*

Question 1 [*6 points total*]

Consider the following LP:

$$
\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^3} & -5x_1 + 8x_2 + 4x_3 \\
\text{s.t.} \quad x_1 + x_2 &= 2 \\
x_2 - x_3 &\leq 3 \\
2x_1 - x_3 &\geq -1 \\
x_1 &\geq 0 \\
x_2 &\in \mathbb{R} \\
x_3 &\leq 0
\end{aligned}
$$

(a) [*1 pts*] Formulate the dual for this LP.

Solution:

$$
\begin{aligned}
\max_{\mathbf{y}\in\mathbb{R}^3} & \; 2y_1 + 3y_2 - y_3 \\
\text{s.t.} \quad y_1 + 2y_3 &\leq -5 \\
y_1 + y_2 &= 8 \\
-y_2 - y_3 &\geq 4 \\
y_1 &\in \mathbb{R} \\
y_2 &\leq 0 \\
y_3 &\geq 0
\end{aligned}
$$

(b) [*2 pts*] Rewrite the primal LP so that constraints are in standard form $\mathbf{Ax} \geq \mathbf{b}$. Vectors may be extended to accommodate slack variables if necessary.

Solution: First, we need to get all variables to be non-negative. We change x_3 to $-x_3$ by flipping its sign everywhere it occurs, and replace x_2 by non-negative slack variables $x_2^+ - x_2^-$. We multiply the $\leq$-constraint by -1, and replace the equality constraint by two inequality constraints $(\leq, \geq)$, the first of which is then multiplied by -1 to change its direction. The primal then becomes

$$
\begin{aligned}
\min_{\mathbf{x}\in\mathbb{R}^4} & -5x_1 + 8x_2^+ - 8x_2^- - 4x_3 \\
\text{s.t.} \quad x_1 + x_2^+ - x_2^- &\geq 2 \\
-x_1 - x_2^+ + x_2^- &\geq -2 \\
-x_2^+ + x_2^- - x_3 &\geq -3 \\
2x_1 + x_3 &\geq -1 \\
\mathbf{x} \geq \mathbf{0}
\end{aligned}
$$

(c) [*2 pts*] If $\mathbf{x}$ is a feasible solution to the primal and the j-th dual constraint is binding, what, if anything, do we know about the primal variable x_j? Justify your answer.

Solution: There are two cases of feasible $\mathbf{x}$. If $\mathbf{x}$ was an optimal solution $\mathbf{x}^*$ to the LP, then the complementary slackness condition would relate the j-th dual constraint to x_j^*. In that case, x_j^* would have to be zero or the constraint had to be binding. Since it is binding, x_j^* is free to take on any value (subject to the primal constraints of course), since complementary slackness is already satisfied by the dual constraint. If, on the other hand, $\mathbf{x}$ was not optimal, complementary slackness would not apply in the first place. In either case, we cannot conclude anything about x_j beyond the primal constraints.

(d) [*1 pts*] Write down the primal coefficient matrix for the original LP. Is this matrix totally unimodular, and why (not)?

Solution: The matrix

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & -1 \\ 2 & 0 & -1 \end{pmatrix}$$

contains a 2, so there exists a submatrix (1×1) with $\det(2) = 2 \notin \{-1, 0, 1\}$, hence the matrix cannot be TUM.

Question 2 [*4 points total*]

You are given an NP-hard minimization problem for which the best known exact algorithm has exponential running time. You therefore attempt to solve it as an ILP using branch-and-bound. Your ILP has polynomially many constraints. To improve the lower bounds obtained by solving the LP relaxations, you use a cutting plane method. You want to avoid Chvátal-Gomory cuts, because you have found a formulation for problem-specific cutting planes you want to use.

(a) [*1 pts*] What is the purpose and effect of improving lower bounds using cutting planes in this setting?

Solution: By finding tighter lower bounds on the solution, it is possible to prune more branches of the tree during the division of the search space. This improves the efficiency of the procedure.

(b) [*2 pts*] Assume you were able to prove that, for any given instance of your problem, there are polynomially many of your cutting planes, and that adding them to its LP relaxation will create an integer polytope as its feasible region, allowing you to solve the ILP optimally using an LP solver. Assuming P $\neq$ NP, what do we know about the separation problem in this case?

Solution: The minimization problem under consideration is NP-hard, and by assumption, P $\neq$ NP. Therefore this problem is not in the complexity class P: there exists no algorithm to solve it in polynomial time.

We have a procedure to solve the problem:

1. Find the (polynomially many) cutting planes, by solving as many instances of the separation problem
2. Add those as constraints to the relaxation of the ILP and solve it

The problem of solving a linear program is in P, so the second step can be done in polynomial time. It is important to note that we added only polynomially many constraints to the LP relaxation, so that the size of the LP is within a polynomial factor of the original problem size.

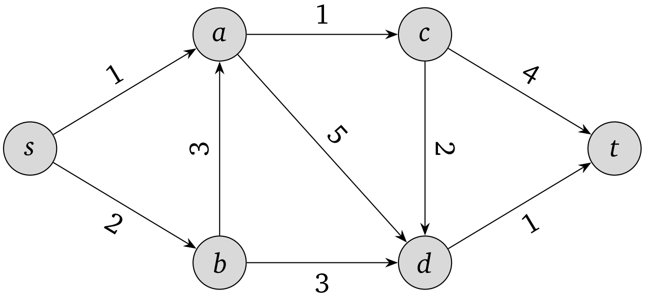
The complexity of the first step depends on the complexity of the separation problem. If the separation problem can be solved in polynomial time, then the first step of the procedure takes polynomial time, and so does the whole procedure. This contradicts the fact that there exists no such procedure. Therefore we can conclude that there exists no polynomial time algorithm to solve the separation problem in this case.

(c) [*1 pts*] Given your answer in (b), is the approach described above a good idea, and why (not)?

Solution: Our approach requires solving multiple instances of a problem for which there is no polynomial time algorithm, which may appear to be a bad idea. However, given the assumption that the minimization problem is itself not in P, we cannot conclude that there exist better algorithms. There may exist a sub-exponential algorithm to solve the separation problem, or an exponential algorithm that performs well on most instances. Therefore the approach described should not be discarded only based on the complexity of the separation problem.

Question 3 [*6 points total*]

Consider the shortest path problem in a weighted, directed graph, i.e. given a (connected) directed graph $G = (V, E)$ with nonnegative cost c(e) for every edge $e \in E$, as well as vertices $s, t \in V$, we want to find a minimum-cost path from s to t in G.



(a) [*2 pts*] Give an integer linear programming formulation of the problem for the graph above. Clearly explain the variables and constraints you are using.

Solution: Introduce a binary variable for each edge (i,j) x_{ij}. x_{ij} is 1 if edge (i,j) is in the shortest path, otherwise is 0.

$$\min_{\mathbf{x}\in\{0,1\}} x_{sa} + 2x_{sb} + 3x_{ab} + x_{ac} + 5x_{ad} + 3x_{bd} + 2x_{cd} + 4x_{ct} + x_{dt}$$

$$\begin{aligned} \text{s.t.} \quad x_{sa} + x_{sb} &= 1 \\ -x_{sa} - x_{ab} + x_{ac} + x_{ad} &= 0 \\ -x_{sb} + x_{ab} + x_{bd} &= 0 \\ -x_{ac} + x_{ct} + x_{cd} &= 0 \\ -x_{ad} - x_{bd} - x_{cd} + x_{dt} &= 0 \\ -x_{ct} - x_{dt} &= -1 \\ \mathbf{x} &\geq 0 \end{aligned}$$

(b) [*2 pts*] Solve the LP relaxation of (a) using CVX. Given that solution, what is the shortest path?

Solution: The LP-relaxation is obtained by removing the requirement for the variables to be binary. The shortest path is $s \to a \to c \to d \to t$ with a total cost of 5. The formulation is given below:

$$\min_{\mathbf{x}\in\mathbb{R}} x_{sa} + 2x_{sb} + 3x_{ab} + x_{ac} + 5x_{ad} + 3x_{bd} + 2x_{cd} + 4x_{ct} + x_{dt}$$

$$\begin{aligned} \text{s.t.} \quad x_{sa} + x_{sb} &= 1 \\ -x_{sa} - x_{ab} + x_{ac} + x_{ad} &= 0 \\ -x_{sb} + x_{ab} + x_{bd} &= 0 \\ -x_{ac} + x_{ct} + x_{cd} &= 0 \\ -x_{ad} - x_{bd} - x_{cd} + x_{dt} &= 0 \\ -x_{ct} - x_{dt} &= -1 \\ \mathbf{x} &\geq 0 \\ \mathbf{x} &\leq 1 \end{aligned}$$

(c) [*1 pts*] Write down the dual of (a).

Solution: For each node i, introduce a dual variable y_i. Then the dual can be written as

$$\begin{aligned} \max_{\mathbf{y}\in\mathbb{R}} \; & y_s - y_t \\ \text{s.t.} \quad & y_s - y_a \le 1 \\ & y_s - y_b \le 2 \\ & y_b - y_a \le 3 \\ & y_a - y_c \le 1 \\ & y_a - y_d \le 5 \\ & y_b - y_d \le 3 \\ & y_c - y_d \le 2 \\ & y_c - y_t \le 4 \\ & y_d - y_t \le 1 \end{aligned}$$

Note that the dual variable $\mathbf{y}$ is unconstrained. Also there are variations of the dual depending on whether you use -1 or $+1$ in the constraint corresponding to the sink node (t) in the primal formulation.

(d) [*1 pts*] Typically, we provide a problem description and ask you to formulate it as an ILP. Now we ask the opposite: given the dual in (c), provide a description of the problem that the dual expresses.

Solution: In a graph G=(V, E) where each edge has a weight associated with it, the variables correspond to nodes in the graph and the constraints correspond to the edges, specifying an upper-bound on the variables associated with their incident nodes. This can be interpreted as follows:

You are given a set of marbles and some pairs of marbles are connected using strings of different lengths. The absolute difference between node variables corresponds to the distance of the two marbles if they are part of the set of taut strings when marbles s and t are pulled as far apart as possible.
Note: There are other correct answers for this question.

Question 4 [*13 points total*]

Suppose you run a store that is open all day and night, all days of the week. You need to have a certain number of personnel in the store at any given time. You know the need of personnel for each 4-hour time slot, see table below:

Time slots	personnel
00:00 – 04:00	5
04:00 – 08:00	7
08:00 – 12:00	12
12:00 – 16:00	8
16:00 – 20:00	12
20:00 – 24:00	9

The union rules do not allow working shifts of 4 hours; if you call in personnel, they always work 8-hour shifts. You would like to minimize the total number of 8-hour shifts.

(a) [*2 pts*] Formulate the problem as an ILP. Clearly explain the variables and constraints you are using.

Solution: Introduce variables x_i for each start of a possible shift. Those variables will indicate the number of personnel that starts their 8-hour shift at 0:00, 4:00 ... 20:00. There will be 6 variables and we require them to be integers. For each time slot, we introduce a constraint saying that the total number of personnel on the slot has to be at least the requested number from the table.

$$\begin{aligned} \min\, & x_1+x_2+x_3+x_4+x_5+x_6 \\ \text{s.t.}\quad & x_1+x_6 \ge 5 \\ & x_1+x_2 \ge 7 \\ & x_2+x_3 \ge 12 \\ & x_3+x_4 \ge 8 \\ & x_4+x_5 \ge 12 \\ & x_5+x_6 \ge 9 \\ & x_i \ge 0, x_i \text{ integer} \end{aligned}$$

(b) [*1 pts*] Write down the dual of the LP relaxation. Explain the notation for the dual variables.

Solution:

$$\begin{aligned} \max\, & 5y_1+7y_2+12y_3+8y_4+12y_5+9y_6 \\ \text{s.t.}\quad & y_1+y_2 \le 1 \\ & y_2+y_3 \le 1 \\ & y_3+y_4 \le 1 \\ & y_4+y_5 \le 1 \\ & y_5+y_6 \le 1 \\ & y_6+y_1 \le 1 \\ & y_i \ge 0 \end{aligned}$$

There is one dual variable for each time slot, with y_j representing the j-th time slot. (They can be seen as an indicator which time slots are covered exactly with the required amount ($y_j = 1$).)

(c) [*2 pts*] Solve the LP-relaxation of the primal and the dual with CVX (include your code and the answer from CVX for objective function value and the variable vectors). What do you observe?

Solution: CVX gives solutions $\mathbf{x} = (2.28, 6.41, 5.56, 4.44, 7.59, 2.72)$ and $\mathbf{y} = (1, 0, 1, 0, 1, 0)$. Since both the LP-relaxation and the dual are LPs, strong duality holds and thus the objective function values are equal ($z = 29$). The primal solution is fractional, but it can easily be seen that the constraint matrix is TU. Thus, the fractional solution indicates that there exist multiple optimal solutions in the primal. (Actually, in this case, simple rounding will give an integer solution with the same objective function value!)

(d) [*2 pts*] Let $\mathbf{x}^*$, $\mathbf{y}^*$ denote primal and dual optimal solutions for the LPs of (c). Write down the complementary slackness conditions.

Solution: For general complementary slackness conditions, see e.g. the lecture notes, section 5.1.

PCS: All primal variables are positive, that is, all dual constraints must hold with equality.

$y_1^* + y_2^* = 1 + 0 = 1$ ok!
$y_2^* + y_3^* = 0 + 1 = 1$ ok! $\ldots$

DCS: The dual variables y_1, y_3, y_5 are positive, that is, the primal constraints one, three and five must hold with equality:

$x_1^* + x_6^* = 2.28 + 2.72 = 5$ ok!
$x_2^* + x_3^* = 6.41 + 5.59 = 12$ ok!
$x_4^* + x_5^* = 4.44 + 7.56 = 12$ ok!

Both the primal and the dual complemenatry slackness conditions are fulfilled.

(e) [*3 pts*] Describe a primal-dual algorithm for the problem, in clear pseudocode.

Solution:

1. Init y_j=0 and $x_i = 0$.
2. Choose a dual constraint that has not yet been processed. All dual constraints are of the form $y_j + y_k \leq 1$, and each dual variable is part of exactly 2 constraints. Set $y_j = 1$ and $y_k = 0$.
3. Let the corresponding primal variable from the first dual constraint be x_i. This variable is part of two constraints. Increase x_i such that the first of the two primal constraints is fulfilled.

4. Loop until primal feasibility is reached.

Note: There are several versions to a PDM for this particular problem. We here show a very simple version that not necessarily gives the best results. Since the dual variable is always part of two constraints that will be fulfilled with equality by raising to 1, this corresponds to two primal variables. It is possible to increase both of these primal variables in the same step. However, it is not so easy to exactly specify how much each of these variables should be increased to still ensure that the entire process will end up in a primal feasible state! A common mistake in f) was to use the primal-dual method for LPs instead of those for ILPs.

(f) [*3 pts*] Solve the problem depicted above using your PDM, and describe what you do in each step.

Solution:

1. Set $y_j = 0$ and $x_i = 0$ for all i, j.
2. Choose the first dual constraint: $y_1 = 1$, this forces $y_2 = 0$. Corresponding primal variable is x_1 and it is part of the primal constraints 1 and 2. Set $x_1 = 5$.
3. Choose the second dual constraint: $y_3 = 1$ (forces $y_4 = 0$) Corresponding primal variable $x_2 = 2$.
4. The third dual constraint is already fulfilled. Primal variable $x_3 = 10$.
5. Choose the fourth dual constraint: $y_5 = 1$ and thus $y_6 = 0$. Primal variable $x_4 = 0$ since $x_3 = 10$ already made sure that this constraint is feasible.
6. The fifth constraint is already fulfilled. Primal variable $x_5 = 12$.
7. The last dual constraint is already fulfilled, and since $x_5 = 12$, we can keep $x_6 = 0$.
8. Both the primal and the dual objective function value are 29. (Note that the complementary slackness conditions are fulfilled as well.)

Question 5 [*6 points total*]

Let $G = (V, E)$ be a complete, undirected graph with node set V and edge set E, such that each edge $e = (u, w)$ has a non-negative weight $c(u, w)$ between any two nodes u, w. We have seen in the lecture that if $c()$ is metric, i.e. obeys the triangle inequality

$$\forall u, v, w \in V : c(u, w) \leq c(u, v) + c(v, w),$$

then the MST heuristic yields a solution H which is a 2-approximation for the shortest round trip H^* in the Traveling Salesperson Problem. In some applications, there actually exists a stronger version of the triangle inequality:

$$\forall u, v, w \in V : c(u, w) \leq \max\left\{c(u, v), c(v, w)\right\}.$$

In this case, the weight function is called *ultrametric*. Show that if the edge weights are ultrametric, the MST heuristic yields an approximate solution H with approximation factor

$$\frac{|V|\,\tilde{c}}{c(\text{MST})}$$

where $\tilde{c}$ is the cost of the heaviest edge in the MST, and $c(\text{MST})$ is the cost of the MST.

Solution: Recall the following facts from the lecture: Removing one edge from H^* creates a spanning tree, which cannot be cheaper than the MST, so $c(\text{MST}) \leq c(H^*)$. Obviously, $c(H^*) \leq c(H)$. The MST heuristic works by first finding an MST, and then traversing “around its edges” (it doubles each edge, creating an Euler cycle). This uses each edge exactly twice, which in total costs $2c(\text{MST})$. To compute H, at each node we either follow an edge or create a shortcut by skipping nodes we have seen before. This creates a Hamilton cycle H. Due to the triangle inequality, the shortcutting edge cannot be longer than the path on the MST that it shortcuts, so $c(H) \leq 2c(\text{MST})$. So we have

$$c(\text{MST}) \leq c(H^*) \leq c(H) \leq 2c(\text{MST}).$$

The worst-case for the approximation ratio $\frac{c(H)}{c(H^*)}$ occurs whenever

$$c(\text{MST}) = c(H^*) < c(H) = 2c(\text{MST}),$$

so

$$\frac{c(H)}{c(H^*)} = \frac{2c(\text{MST})}{c(\text{MST})} = 2.$$

For the ultrametric, the claim is that the approximation ratio is

$$\frac{c(H)}{c(H^*)} = \frac{|V|\,\tilde{c}}{c(\text{MST})},$$

so we need to show that

$$c(H) \leq |V|\,\tilde{c}.$$

Now let’s see what’s different for an ultrametric as compared to a metric when applying the MST heuristic. As before, at each node, you decide whether to follow an MST edge or create a shortcut edge. For an ultrametric, the shortcut edge is not just at most as long as the sum of edges on the path it shortcuts, but in fact at most as long as the longest edge on that path! This follow from a simple induction argument (the intuition given here suffices, we did not expect to see a formal proof): by definition, if we shortcut a path $P_1^3 := (v_1, v_2, v_3)$ of 3 nodes, we have

$$c(v_1, v_3) \leq \max\left\{c(v_1, v_2), c(v_2, v_3)\right\}.$$

To shortcut $P_1^4 := (v_1, v_2, v_3, v_4)$ using (v_1, v_4), we know the cost can be at most the maximum of the two edges (v_1, v_3) and (v_3, v_4), so

$$c(v_1, v_4) \leq \max\left\{c(v_1, v_3), c(v_3, v_4)\right\}.$$

$$\begin{aligned}
&\leq \max\left\{\max\left\{c(v_1, v_2), c(v_2, v_3)\right\}, c(v_3, v_4)\right\} \\
&= \max\left\{c(v_1, v_2), c(v_2, v_3), c(v_3, v_4)\right\} \\
&= \max_{1 \leq i < 4} c(v_i, v_{i+1}).
\end{aligned}$$

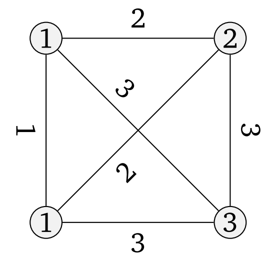
In general,

$$\begin{aligned}
c(v_1, v_k) &\leq \max\left\{c(v_1, v_{k-1}), c(v_{k-1}, v_k)\right\} \\
&\leq \max_{1 \leq i < k} c(v_i, v_{i+1})
\end{aligned}$$

So for each of the $|V|$ nodes, we either pick an MST edge, which costs at most $\tilde{c}$ (the cost of the longest edge in the entire MST), or a shortcut edge which costs at most as much as the longest edge on the path in the MST it shortcuts, which in turn also costs at most $\tilde{c}$. Therefore,

$$c(H) \leq |V|\,\tilde{c}.$$

Remark: Some students based their argument on the claim that the graph cannot have more than two different edge costs due to ultrametricity. This is not true, as the following counterexample shows: Take any complete graph and label its nodes with arbitrary weights. Let the edge weight be the maximum weight of its incident nodes. The resulting graph will have ultrametric edges, e.g.


1
2
1
3
2
3
3
1
2
3