1 Exercise 1

Bianchini - Evangelisti

1.a We have `m` elves to deliver `n` (m<n) presents to `n` houses. Each elf `e` has a set of doable houses H_e to which he can deliver, and each delivery takes an elf one hour to be completed. We want to deliver all presents minimizing the time. In order to define the problem as an ILP we have defined the following binary variable: $X_{e,h} = \begin{cases} 1 & \text{if elf e delivers to house h} \\ 0 & \text{otherwise} \end{cases}$

The problem, with constraints and objective function, is defined as it follows:

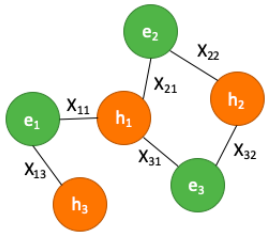
min(T)	Objective function
s.t.	
$\sum_{e \in E} X_{e,h} = 1 \quad \forall h \in H_e$	different elves don't deliver to the same house
$\sum_{e \in E, h \in H_e} X_{e,h} = n$	all n presents are delivered
$X_{e,h} = 0 \quad \forall e \in E$ and $h \notin H_e$	each elf doesn't deliver to a not doable house
$X_{e,h} \in \{0,1\} \quad \forall e \in E$ and $h \in H_e$	Integrity of the binary variables $X_{e,h}$
$\sum_{h \in H_e} X_{e,h} \leq T \quad \forall e \in E$	T is the time until the last elf is finished[1]

1 Considering $T[e]$ the time taken by elf e to do his deliveries (number of houses he do), $T = max(T[e] \quad \forall e \in E$

In order to relax the previous problem to an LP, we remove the constraint on the integrity of the binary variables $X_{e,h}$ and we insert: $0 \leq X_{e,h} \leq 1 \quad \forall e \in E$ and $h \in H_e$.

1.b The following algorithm takes as input an optimal solution to the LP problem explained in 1.a and outputs a feasible solution to Santa's problem.

The solution of the LP is given by the values $X_{e,h}$ $\forall e \in E$ and $h \in H$, this could be represented in a graph structure where each edge corresponds to a variable $X_{i,j} > 0$.



For each house $h \in H$:

- we choose an elf e randomly (with probability equal to $X_{e,h}$[1]) among the outer edges from the house h, and we set $X_{e,h} = 1$;
- we set the value of all the other out edges from h to 0.

This algorithm outputs a feasible solution, in fact: we made the variables $X_{e,h} \in \{0,1\}$; due to the fact that for each house h we set $X_{e,h} = 1$ and $X_{i,j} = 0 \quad \forall i \in E$ s.t. $i \neq e$ and $j = h$ the first constraint is respected; the second one comes from the previous considering that we have `n` houses; all the others are respected because the input was a solution for LP.

The algorithm runs in polynomial time, in particular in $O(nm)$: because we cycle over all the n houses and each house is connected to at most m elves.

Assuming X_e the number of houses visited by elf e, the expected value of X_e is

$$E[X_e] = \sum_{h \in H} X_{e,h} P(X_{e,h}) \leq \sum_{h \in H} P(X_{e,h}) = \sum_{h \in H} X^{LP}_{e,h} \leq T^{LP} = OPT_{LP} \quad \forall e \in E$$

considering that the probability $P(X_{i,j})$ $\forall i \in E, j \in H$ is equal to the solution $X_{i,j}$ of the LP. Since LP has fewer constraints than ILP formulation we have that: $OPT_{LP} \leq E[OPT*] \leq OPT_{LP}$, so the optimal solution of this algorithm OPT* is, in expectation, just as good as the optimal of LP.

1.c Considering X_e as previously defined, $\mu = E[X_e] = OPT_{LP}$, we apply the Chernoff's Bound $P[X \geq (1+\epsilon)\mu] < exp(-\mu\ min\{\epsilon, \epsilon^2\}/3)$ with $\epsilon = 3(\frac{ln(m^2)}{OPT_{LP}} + 1)$.

Since $\epsilon \geq 1$, $min\{\epsilon, \epsilon^2\} = \epsilon$ so:

$$P[X_e \geq (4 \cdot OPT_{LP} + 6 \cdot ln(m))] < exp(-OPT_{LP}(\frac{ln(m^2)}{OPT_{LP}} + 1)) = exp(-OPT_{LP} - ln(m^2)) = \frac{1}{m^2} \cdot \frac{1}{e^{OPT_{LP}}} \leq \frac{1}{m^2}$$

Applying the Union Bound we have that with probability $\geq 1 - \frac{1}{m}$ no elf delivers to more than $O(OPT_{LP} + ln(m))$ houses, that, since each delivery takes 1 h, is an upper bound on the finishing time.

[1] solution of the LP problem which can have a value $0 \leq v \leq 1$

2 Exercise 2

Bianchini - Evangelisti

The problem consists in creating 187 different sleights, having available a set P of n different pieces, respecting some Santa's constraints. We can formalize our problem as it follows.
We can number the n different pieces from 1 to n, where X_i is the i-th piece, and the sleights numbered from 1 to 187.
We can define: $X_{i,j} = \begin{cases} 1 & \text{if the piece i is in the sleight j} \\ 0 & \text{if the piece i is NOT in the sleight j} \end{cases}$
Each Santa's constraint can be represented as a Boolean expression of literals $X_{i,j}$ using operators AND, OR, NOT, XOR. For example, if we have a constraint as "*each sleight needs a front light*" and we have the set $L = \{X_3, X_5, X_9\}$ of all front lights, we would insert the following clauses $\forall j \in \{1, 187\}$: $(X_{3,j} \vee X_{5,j} \vee X_{9,j})$ where each one of this clause is in conjunction.
Since we have to build 187 different sleights we need to add to the previously defined Santa's constraints some extra clauses to ensure that for any possible combination of two sleights they have at least one piece different from the other. Taking two sleights k,l ($k, l \in \{1, 187\}$ with $k \neq l$) we can say that they're different if $S_{k,l} = 1$, where:

$$S_{k,l} = (X_{1,k} \oplus X_{1,l}) \vee (X_{2,k} \oplus X_{2,l}) \vee ... \vee (X_{n,k} \oplus X_{n,l})$$

so, to say that sleights are all different we add the conjunction of $S_{k,l}$ clauses $\forall k, l \in \{1, 187\}$ with $k \neq l$.
If all constraints are satisfied, each sleight s will be constructed with all the pieces such that $X_{i,s} = 1$ $\forall i \in$\{1,n\} $\forall s \in \{1, 187\}$.
This problem is $\in$ NP-Hard because an instance of SAT problem can be represented as an instance of our problem: **SAT $\leq_P$ Santa's problem**.
Having an instance of SAT, each literal X_i in the SAT formulation can be represented as $X_i = \bigvee_{\forall j \in \{1,187\}} X_{i,j}$ and each clause remains the same with the substitution in which each literal becomes the disjunctive sub-clause written before. The number of different literals in SAT is the equivalent of the number of pieces that we have in Santa's problem. So we add to the clauses representing the literals the clauses ensuring that the resulting sleights will be different, so: $\bigwedge_{\forall k,l \in \{1,187\} \text{ with } k \neq l} S_{k,l}$.
For example SAT's instance $(X_1 \vee \neg(X_2) \vee X_3) \wedge (\neg(X_1))$ becomes:

$$((X_{1,1} \vee ... \vee X_{1,187}) \vee \neg(X_{2,1} \vee ... \vee X_{2,187}) \vee (X_{3,1} \vee ... \vee X_{3,187}) \wedge (\neg(X_{1,1} \vee ... \vee X_{1,187})) \bigwedge_{\forall k,l \in \{1,187\} \text{ with } k \neq l} S_{k,l}$$

Since the first constraints in Santa's formulation are the ones in SAT's instance, if there's a solution for Santa's problem, also SAT has a solution.
In order to say that if Santa's problem has not a solution, neither SAT has it, we have to consider that: independently from SAT's instance, in order to build 187 sleights we need at least 8 different pieces so that $2^8 = 256$ represent all the possible combinations. So, in order to use the previous reduction, we need to add 8 auxiliary pieces so that, if the constrains coming from SAT are satisfied also the difference between the sleights would be satisfied.
Considering m the number of the literals in SAT's instance, we can define the auxiliary variables as:
$Y_{i,j} = \begin{cases} 1 & \text{if the auxiliary piece i is in the sleight j} \\ 0 & \text{otherwise} \end{cases}$ $\quad \forall i \in \{m+1, m+8\}$ and $j \in \{1, 187\}$
In that case, $S_{k,l}$ becomes:

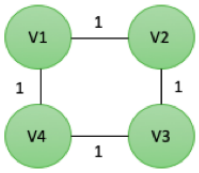
$$S_{k,l} = (X_{1,k} \oplus X_{1,l}) \vee (X_{2,k} \oplus X_{2,l}) \vee ... \vee (X_{m,k} \oplus X_{m,l}) \vee (Y_{m+1,k} \oplus Y_{m+8,l}) ... \vee (Y_{m+8,k} \oplus Y_{m+7,l})$$

So the auxiliary variables $Y_{i,j}$ don't influence the SAT's constraints and we can say that if we have a solution that satisfies Santa's problem, it will be satisfied also SAT one, while if Santa's problem is not satisfiable, it depends only on the SAT's constraints, so it means that neither SAT problem is satisfiable.
So, we can say that Santa's problem is $\in$ NP-Hard because, if we could have a polynomial algorithm that is able to solve Santa, it would also be able to solve SAT in poly-time, but since SAT is NP-Hard, for the assumption that $P \neq NP$, this is impossible.

3 Exercise 3

Bianchini - Evangelisti

3.a We have a game in which each player i has a Strategy space $\alpha_i = \{LEFT, RIGHT\}$ and $\alpha = (\alpha_1, ..., \alpha_n)$. We represent it with an undirected graph $G = (V, E)$ in which each player controls a vertex. Defining as N_i the neighbours of a vertex i, the payoff of player i is: $u_i(\alpha) = \sum_{e \in CUT(\alpha) \cap N_i} w_e$ where w_e (>0) is the weight of the vertex e and $CUT(\alpha)$ is the set of edges that connects the right side with the left one in the α configuration.
We've designed a cut game with Price of Anarchy equal to 2, with 4 players, each one associated to a vertex in the graph presented in the following Figure, where the payoff values are inserted in a table, in which the equilibria are the orange cells.



V3, V4 \ V1, V2	L, L	L, R	R, L	R, R
L, L	0, 0, 0, 0	1, 2, 1, 0	2, 1, 0, 1	1, 1, 1,1
L, R	1, 0, 1, 2	2, 2, 2, 2	1, 1, 1, 1	0, 1, 2, 1
R, L	0, 1, 2, 1	1, 1, 1, 1	2, 2, 2, 2	1, 0, 1, 2
R, R	1, 1, 1, 1	2, 1, 0, 1	1, 2, 1, 0	0, 0, 0, 0

In each cell the values are (V1, V2, V3, V4).

Having the set of states $S = \alpha_1 \times \alpha_2 \times \alpha_3 \times \alpha_4$, the set of the equilibria E and the set of players $P = \{1, 2, 3, 4\}$, the price of anarchy is equal to:

$$\frac{max_{\alpha \in S} \sum_{i \in P} u_i(\alpha)}{min_{\alpha \in E} \sum_{i \in P} u_i(\alpha)} = \frac{8}{4} = 2$$

3.b In order to prove that the Price of Anarchy of cut games is at most 2 in the general case, we have to prove that

$$\frac{max_{\alpha \in S} \sum_{i \in P} u_i(\alpha)}{min_{\alpha \in E} \sum_{i \in P} u_i(\alpha)} \leq 2$$

We can observe that the $max_{\alpha \in S} \sum_{i \in P} u_i(\alpha)$ is when a player i chooses a side and all its neighbours choose the other side, because in that case he'd have the payoff equal to the sum of all his edges N_i:

$$max_{\alpha \in S} \sum_{i \in P} u_i(\alpha) = \sum_{i \in P} \sum_{e \in N_i} w_e \quad \text{, so, in general:} \quad max_{\alpha \in S} \sum_{i \in P} u_i(\alpha) \leq \sum_{i \in P} \sum_{e \in N_i} w_e$$

In a Pure Nash Equilibrium the total weight of the neighbours N_i of a vertex i in the cut is at least half the total weight of all the neighbours **(1)**:

$$\sum_{e \in CUT(\alpha) \cap N_i} w_e \geq \frac{1}{2} \sum_{e \in N_i} w_e \quad \text{, so, in general:} \quad min_{\alpha \in E} \sum_{i \in P} u_i(\alpha) \geq \frac{1}{2} \sum_{i \in P} \sum_{e \in N_i} w_e$$

Then, the price of anarchy is:

$$\frac{max_{\alpha \in S} \sum_{i \in P} u_i(\alpha)}{min_{\alpha \in E} \sum_{i \in P} u_i(\alpha)} \leq \frac{\sum_{i \in P} \sum_{e \in N_i} w_e}{\frac{1}{2} \sum_{i \in P} \sum_{e \in N_i} w_e} \leq 2$$

Having that in a Pure Nash Equilibrium $u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i}) \quad \forall i \in P, \forall s \in S$. We can prove the sentence (1) by contradiction assuming that we have a PNE and exists a player i for which $\sum_{e \in CUT(\alpha) \cap N_i} w_e < \frac{1}{2} \sum_{e \in N_i} w_e$. Note that in a PNE the player, knowing the choices of the others, chooses its best strategy possible, but in that case if we change strategy we obtain:

$$u'_i = \sum_{e \in N_i} w_e - \sum_{e \in CUT(\alpha) \cap N_i} w_e = \frac{1}{2} \sum_{e \in N_i} w_e + \frac{1}{2} \sum_{e \in N_i} w_e - \sum_{e \in CUT(\alpha) \cap N_i} w_e = \frac{1}{2} \sum_{e \in N_i} w_e + \Omega$$

Since $\Omega > 0$ for the initial assumption, player i would have a better strategy to choose, so we're not in a Nash Equilibrium and (1) is proved.

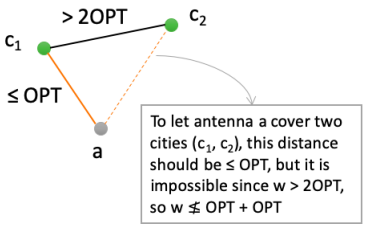
4 Exercise 4

Bianchini - Evangelisti

We have a complete graph $G = (V, E)$, the set A of possible antennae location and the set S of cities ($V = A \cup S$), we can define a distance function (dist(x,y)) as the weight of the edge connecting vertices x,y. We know that the weights in the graph satisfy the triangle inequality. We have to find a subset of k antennae $U \subset A$ s.t. $max_{x \in S}\ (min_{y \in U} d(x, y))\ x, y \in V$ is minimized.

4.2 In order to find a 3-approximation algorithm, we apply *vertex k-Center* to the cities' set S, and we obtain a subset of S, $S' = \{c_1, c_2, ..., c_k\}$ of k cities. In order to select k antennae, we choose $U = \{a_i \quad \text{s.t} \quad dist(a_i, c_i) = min_{a \in A}(dist(a, c_i)) \quad | \quad c_i \in S'\}$, so, for each c_i in S', we're selecting the nearest antenna (if more, we select one).
We consider OPT as the result of $max_{x \in S}\ (min_{y \in U} d(x, y))\ x, y \in V$ where U is an optimal solution for our problem. We can say that $dist(c_i, a_i) \leq OPT$, for each $c_i \in S$, where a_i is c_i's nearest antenna. Note that if the number of cities is equal to k, we can easily choose for each city its nearest antenna and the solution is optimal. If $|S| = m$ and $m < k$, we select the m-nearest and other $k - m$ antennae that are the nearest if not considering the ones already taken.
So, we have to analyze the case in which the number of cities is greater than k and prove the 3-approximation. We choose an arbitrary city $c \in S$ and we obtain 2 different cases:
• $\exists\ c_i \in S'$ s.t. $dist(c, c_i) \leq 2OPT$, then $dist(c, f_i) \leq dist(c, c_i) + dist(c_i, f_i) \leq 2OPT + OPT = 3OPT$ that is always true for the triangle inequality;

• $dist(c, c_i) > 2OPT$ for each $c_i \in S'$, then, by the construction of the optimal solution, also the distance between any $c_i, c_j \in S'$, where $j \neq i$, is greater than $2OPT$; because, if that was not true, c would be $\in S'$ instead of c_i or c_j. If that is the case, we have $S' \cup \{c\}$ as a set of cardinality $k + 1$ where for each couple x, y in the set, $dist(x, y) > 2OPT$. The problem is that we need $k + 1$ antennae to cover this cities so OPT could not be obtained from a solution made of k antennae.
So the first case is the only possible and we obtain that the optimum of the approximation algorithm is $= 3OPT$, so this is a 3-approximation.



4.3 In order to show that finding an α-approximation with $\alpha < 3$ is NP-hard, we use the reduction with Dominating Set: DS $\leq_P$ k-antennae. Given a graph $G = (V, E)$ representing an instance of DS, we can define an instance of our problem as:
$\forall n \in V$ we have two vertices c_n, a_n and we set the distances (respecting the triangle inequality) as it follows: $dist(x_i, y_j) = \begin{cases} 2 & \text{if}(x_i, y_j) \in A \text{ or } (x_i, y_j) \in S \\ 1 & \text{if } i = j \text{ or } (x_i, y_j) \in E \\ 3 & \text{otherwise} \end{cases}$
If the Dominating Set of k dimension exists, then taking as a solution of our problem the k antennae coming from the k vertices that are in the DS solution and the OPT value is equal to 1. Otherwise if that solution doesn't exist we obtain an OPT value equal to 3. Considering the other direction, we can say that if we have a value 1 solution made of k-antennae for our problem, the k vertices associated to the k antennae can be chosen as Dominating Set.

If we could have and α-approximation algorithm ($\alpha < 3$) for our problem, given the previously defined reduction, we could use this algorithm also to solve DS problem, since $\exists$ DS with k vertices iff k-antennae has value equal to 1, otherwise if k-antennae has value 3, k-DS doesn't exist. So, given a graph G=(V,E) we can compute the k-DS reducing it to a k-antennae problem and applying the α-approximation algorithm that will return a value $\leq \alpha OPT$. So if the dominating set exist it returns $\alpha < 3$, otherwise it would return $3\alpha \geq 3$. So, analyzing if the output of this algorithm is greater or smaller than 3 we can know if the k-dominating set exists or not. But, since it doesn't exist a poly-time algorithm that solves the DS problem, because it is a NP-Hard problem, we can say that finding an α-approximation algorithm for k-antennae problem, with $\alpha < 3$, is a NP-Hard problem.