Exercise 1

Part 1

Let $x_{e,h}$ be a binary variable that tells us if the elf $e \in E$ will deliver a present to the house $h \in H$. If e cannot deliver to h by specification, we simply put $x_{e,h} = 0$. The elves work in parallel, therefore we have to minimize $\max_{e \in E} \sum_{h \in H} x_{e,h}$. This is not a linear function, so we add an extra variable M in the ILP formulation

$$
\begin{aligned}
\min \quad & M \\
\text{s.t.} \quad & M \geq \sum_{h \in H} x_{e,h} && \forall\, e \in E \\
& \sum_{e \in E} x_{e,h} \geq 1 && \forall\, h \in H \\
& \sum_{h \in H} x_{e,h} \leq |H_e| && \forall\, e \in E \\
& x_{e,h} \in \{0, 1\}
\end{aligned}
$$

The LP formulation is given relaxing the binary variable $x_{e,h} \in [0, 1]$.

Part 2

$\forall\, e, h$, $x^*_{e,h}$ is the LP optimal solution. A solution to Santa's problem can be obtained setting $x_{e,h}$ to 1 with probability $x^*_{e,h}$ and 0 with probability $1 - x^*_{e,h}$. This solution is feasible because it respects all the constraints. We can compute the expected value $E[x_{e,h}] = x^*_{e,h}$. By linearity,

$$
E\left[\sum_{h \in H} x_{e,h}\right] = \sum_{h \in H} x^*_{e,h} \leq \max_{e \in E} \sum_{h \in H} x^*_{e,h} = OPT
$$

This means that, in expectation, the solution is as good as the LP optimum.

Part 3

We want to prove that $P(\sum_{h \in H} x_{e,h} \leq O(\log m + OPT)) \geq 1 - 1/m$.

$$
P\left(\sum_{h \in H} x_{e,h} \geq (1 + \epsilon)\mu\right) < exp\left(-\mu \frac{\min\{\epsilon, \epsilon^2\}}{3}\right)
$$

$$
exp\left(-\mu \frac{\min\{\epsilon, \epsilon^2\}}{3}\right) = \frac{1}{m} \Rightarrow -\mu \frac{\min\{\epsilon, \epsilon^2\}}{3} = -\ln m \Rightarrow \epsilon = 3\frac{\ln m}{\mu}
$$

$$
P\left(\sum_{h \in H} x_{e,h} \geq \left(1 + 3\frac{\ln m}{\mu}\right)\mu\right) < \frac{1}{m} \Rightarrow P\left(\sum_{h \in H} x_{e,h} \geq O(\log m + OPT)\right) < \frac{1}{m}
$$

Exercise 2

First, we consider the problem of making a single sleigh respecting all the restrictions imposed by Santa. The problem is to output a sequence of usable pieces $p_{i_1}, p_{i_2}, ..., p_{i_{k_i}} \in P$ given a set of formal constraints. We can simplify the notation by using a boolean variable $p_j \in \{0, 1\}$, that tells us if a certain piece is used for the sleigh, and the tuple $(p_1, p_2, ..., p_n) \in \{0, 1\}^n$ to show the composition of the sleigh. Our task can be simply seen as a constraint satisfaction problem: $P = \{p_1, ..., p_n\}$ is the set of variables, $\{0, 1\}$ is the domain of each variable, and the constraints are those given by Santa's input. In particular, a formal constraint specifies a relation over a subset of variables $C_i \subseteq \{0, 1\}^k$, that is all the values allowed for those variables. If the use of p_{22} implies the use of p_{37}, we can write $\neg p_{22} \vee p_{37}$: all the tuples are allowed except the one with $p_{22} = 1$ and $p_{37} = 0$, that would make the previous formula false.

One of the well-known constraint satisfaction problem is SAT, the problem of finding an assignment that satisfies a CNF logic formula Φ. SAT is an NP-complete problem, and this means that if we reduce it to another problem X (SAT $\leq_p$ X), X is at least as hard as the hardest NP problem. This is what we can use to prove that Santa's request is hard in the general case. Given a set of literals, a CNF formula is $\Phi = \bigwedge_{i=1}^{m} C_i$, where $C_i = \bigvee_{j=1}^{n} x_j$. We can reduce it as follows: there is a variable (piece of sleigh) p_j for each boolean variable x_j in the clauses, and there is a constraint for each clause C_i. Santa's problem has a solution if and only if Φ is satisfiable: the assignment to the variables of the formula can stand for which pieces to take in the making of the sleigh, and vice versa. Furthermore, we can prove that Santa's problem is in NP: having a list of used pieces (certificate), we can check in polynomial time whether it satisfies all the given constraints (certifier). It means that the 1-sleigh version of our problem is NP-complete.

The last step of our proof is to show that the 187-sleigh version of the problem, as an obvious NP problem, encounters the same difficulties as the previous one. In this case, we need 187 tuples that satisfy the constraints and are different for at least one value. We can see it as repeating 187 times the same problem: we solve the 1-sleigh version and add a new constraint each time to prevents the picking of the already-chosen tuples. This constructive method helps us to understand how many different sleighs we can make without violating any restrictions. Therefore, the 187-sleigh version is at least as hard as one of its iteration, and it means that it is NP-complete, too. If we can solve it in polynomial time, we can do the same thing with the 1-sleigh version of Santa's problem.

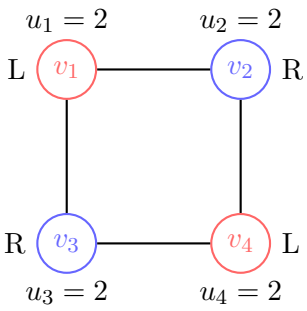
Exercise 3

Part 1

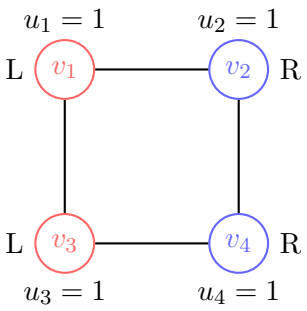
Given the set of states $S = \alpha_i^{|V|}$ and the set of equilibria $E \subseteq S$, the Price of Anarchy is

$$PoA = \frac{\max_{\alpha \in S} \sum_{i \in V} u_i(\alpha)}{\min_{\alpha \in E} \sum_{i \in V} u_i(\alpha)} \quad \text{with } u_i(\alpha) = \sum_{e \in CUT(\alpha) \cap N_i} w_e$$

This means that we have to find the state of the game that gives us the higher social utility and the equilibrium state that gives us the lowest social utility. We can design an instance of the cut game as follows



(a) $\max_{\alpha \in S} \sum_{i \in V} u_i(\alpha) = 8$



(b) $\min_{\alpha \in E} \sum_{i \in V} u_i(\alpha) = 4$

The players associated with the red vertices choose the strategy LEFT, while the players associated with the blue vertices choose the strategy RIGHT. In the described instance the Price of Anarchy is $PoA = 8/4 = 2$.

Part 2

The total weight of the edges in the intersection $CUT(\alpha) \cap N_i$ is at most the total weight of the edges of a vertex v_i (because $CUT(\alpha) \cap N_i \subseteq N_i$)

$$u_i(\alpha) = \sum_{e \in CUT(\alpha) \cap N_i} w_e \leq \sum_{e \in N_i} w_e$$

Furthermore, if a strategy profile α is a Nash equilibrium we know that

$$\sum_{e \in CUT(\alpha) \cap N_i} w_e \geq \sum_{e \in N_i \setminus CUT(\alpha)} w_e$$

$$\sum_{e \in N_i} w_e \leq 2 \sum_{e \in CUT(\alpha) \cap N_i} w_e \quad \Rightarrow \quad \sum_{e \in CUT(\alpha) \cap N_i} w_e \geq \frac{1}{2} \sum_{e \in N_i} w_e$$

Now we can prove that the Price of Anarchy is at most 2

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

Exercise 4

Part 2

The given problem has many similarities with the center selection problem, e.g., the distance function, its properties and the objective function to be optimized. In this case, the acceptable centers are the set of antennae $A \subset V$. The algorithm needed to output the set $U \subset A$ proceeds in the following way: it takes the city c_i farthest from any already-chosen antenna, that is the one with maximum distance $d(c_i, U) = \min_{y \in U} d(c_i, y)$, and then it picks the closest antenna to the city c_i. The algorithm stops when $|U| = k$ and it can be proved that this is a 3-approximation.

Let c_i and a_i be a city and its closest antenna chosen by the described algorithm. a_i^* is the antenna nearest to c_i in the optimal solution. If $r(U) = \max_{x \in S} \min_{y \in U} d(x, y)$ is the covering radius, we know by definition that $d(c_i, a_i^*) \leq r(U^*)$. If c_k is an other city such that $d(c_k, a_i^*) \leq r(U^*)$, we want to prove that c_k is not so far from the antenna in our solution a_i. Due to the fact that a_i is the closest antenna to c_i, it holds that $d(c_i, a_i) \leq r(U^*)$, and by triangle inequality $d(a_i, a_i^*) \leq d(c_i, a_i) + d(c_i, a_i^*) \leq 2r(U^*)$. Now we can use the same property and we obtain $d(c_k, a_i) \leq d(c_k, a_i^*) + d(a_i^*, a_i) \leq r(U^*) + 2r(U^*) = 3r(U^*)$

Part 3

We have to reduce an NP-hard problem to an instance of our problem and show that, if there was a better approximation, we would solve the former problem in polynomial time. In order to choose the NP-hard problem, we note that the set of vertex V is partitioned into two subsets A and S. This means that it can be simpler to reduce one problem with the same property.

The set cover problem is the one we need. Given a set U of elements, a collection $S_1, S_2, ..., S_m$ of subsets of U, and an integer k, the task is to find a collection of size less than or equal to k whose union is equal to U. An arbitrary instance can be reduced to our problem: the set U of elements is the set S of cities and the collection of subsets is the set A of antennae, i.e. for each subset in the collection there is an antenna $a \in A$. Now we set the distances: we'll have $d(c, a) = 1$ if the subset related to a contains the element related to c, and $d(c, a) = 3$ otherwise. The distances between two cities or two antennae can be 2 because of the triangle inequality.

The set cover problem has a solution of size k if and only if the created instance has an optimal solution such that $|U^*| = k$ and $r(U^*) = 1$. It is simple to see that if we have k subsets whose union of elements gives us the universe set, then k antennae are linked to the entire set of cities with a distance equal to 1, and vice versa. If we had a $(3 - \epsilon)$-approximation, than we could obtain a solution $r(U) \leq 3 - \epsilon$ or $r(U) = 3$ and solve the set cover problem in polynomial time.