Algorithm Design
Homework 2

Brunetti Jacopo 1856271
Carmignani Federico 1845479

January 17, 2022

1 Exercise 1

Brunetti, Carmignani

1.a) The algorithm

The problem proposed tells a story in which Santa is worried about his employee relations and he wants to solve this issue according to each elf has a set of elves that are his friends, and Santa, in order to make sure that the elves are happy, wants to choose some elves which become complaint officers, who will have the task to report to him any worries or complaints; so, each complaint officer reports to Santa his complaints and his friends' complaints.
The algorithm that it is necessary to define has to find **exactly *k* elves** that work as complaint officers such that as many elves as possible have at least one friend that is a complaint officer.
It is possible to model this problem as follows: create several sets, one for each elf, and a number k, the goal is to select k of these sets such that the maximum number of elements, joining all elements among all the sets without repetitions, are covered. In order to do this, each set S_i is the set of elves of which the i^{th} elf gathers worries and k is the maximum number of complaint officers to reach defined by the problem. Each set S_i is composed by the elf i^{th}, to whom the set is referred, and all the elves that are his friends.
The algorithm takes as inputs k and a collection of sets $S = S_1, \ldots, S_m$, where m is the cardinality of the set of elves W, and it finds a subset $S' \subseteq S$ of sets such that $|S'| = k$ and the number of covered elements $| \cup_{S_i \in S'} S_i|$ is maximized.
The **greedy** algorithm for this problem is very intuitive and relies on the idea according to which, for each step of the algorithm, a greedy choice is done: **the set which contains the largest number of uncovered elements is taken**.

Algorithm 1 Greedy

```
Require: W ⇐ set of elves, S ⇐ collection of sets, each one related to an elf, containing himself and his friends.

  procedure Ex_1($W$, $S$):
      while less than $k$ sets are taken do
          Pick the set that covers the maximum number of uncovered elements.
          Mark the elements in the chosen set as covered.
      end while
```

At the end of this algorithm, k sets are taken and each set S_i taken defines the choice of the elf i^{th}, appointing, consequently, ***k* elves that maximize the total number of elves having at least one of their friends among the compliant officers**.

1.b) The proof of the maximum approximation ratio $(\mathbf{1} - \frac{\mathbf{1}}{\mathbf{e}})$

It is possible to prove that, for large numbers of k, **the algorithm approximates a solution with ratio no more than** $(1 - \frac{1}{e})$.
In the theory of approximation algorithms, APX is defined as the class of problems P such that, for some $c \leq 1$, there exists a polynomial-time c-approximation algorithm for P. This problem belongs to it and it is asked to demonstrate that this value c must be $(1 - \frac{1}{e})$. An approximation algorithm returns a solution that is guaranteed to be close to the optimal one, in this case, defining SOL as the value of the solution of the greedy algorithm and OPT as the value of an optimal solution, the goal, being **a maximization problem**, is that:

$$\mathbf{SOL} \geq \mathbf{c} \cdot \mathbf{OPT} = (\mathbf{1} - \tfrac{\mathbf{1}}{\mathbf{e}}) \cdot \mathbf{OPT}.$$

It is possible to denote new_elves_i as the number of the new elves covered by the greedy algorithm at i^{th} step, when it takes the i^{th} set. Instead, tot_elves_i is the total number of elves covered up to the i^{th} step, so $tot\_elves_i = \sum_{j=i}^{i} new\_elves_j$.
The value of tot_elves at k^{th} step, tot_elves_k, represents the final number of covered elves and it has been defined as the SOL value of the problem.
Let $uncovered\_elves_i$ as the number of uncovered elements after the i^{th} step, it can be also seen as the difference between the optimal value and the total number of covered elves at i^{th} step, so $uncovered\_elves_i = OPT - tot\_elves_i$.
It can be said that the number of the new covered elves at the $(i+1)^{th}$ step is always greater or equal than the k^{th} part of the uncovered elements after i^{th} step: $new\_elves_{i+1} \geq \frac{uncovered\_elves_i}{k}$.
This assumption can be done knowing that the optimal solution covers OPT elements at the k^{th} step and it means that, for each iteration, there is at least a set whose size is greater or equal than the k^{th} part of the remaining uncovered elements, $(\frac{uncovered_elves_i}{k})$, so using a greedy algorithm which chooses always the set that covers the maximum number of uncovered elements, the chosen set at each iteration should be at least the k^{th} part of remaining uncovered elements: $new\_elves_{i+1} \geq \frac{uncovered\_elves_i}{k}$.
Furthermore it is possible to say that $uncovered\_elves_{i+1} \leq (1 - \frac{1}{k})^{i+1} \cdot OPT$. It is easy to prove this **by induction**: it is trivial to show for i=0 since, according to what is told before, the number of elves not covered after the first iteration is at most equal to $OPT - \frac{OPT}{k}$; it follows that, for the inductive hypothesis, it can be assumed that $uncovered\_elves_i \leq (1 - \frac{1}{k})^i \cdot OPT$, so we want to show that $uncovered\_elves_{i+1} \leq uncovered\_elves_i - new\_elves_{i+1}$. Then, by previous assumption, $uncovered\_elves_{i+1} \leq uncovered\_elves_i - \frac{uncovered\_elves_i}{k} = uncovered\_elves_i \cdot (1 - \frac{1}{k})$. By inductive hypothesis, $uncovered\_elves_i \cdot (1 - \frac{1}{k}) \leq (1 - \frac{1}{k})^i \cdot OPT \cdot (1 - \frac{1}{k}) = (1 - \frac{1}{k})^{i+1} \cdot OPT$.
So, it is proved that $uncovered\_elves_{i+1} \leq (1 - \frac{1}{k})^{i+1} \cdot OPT$.
Using the previous assumptions, it can be said that at k^{th} step $uncovered\_elves_k \leq (1 - \frac{1}{k})^k \cdot OPT$ and, for large numbers of k, it is mathematically true that $(1 - \frac{1}{k})^k \to \frac{1}{e}$, so $uncovered\_elves_k \leq (1 - \frac{1}{k})^k \cdot OPT \leq \frac{OPT}{e}$. Now it is possible to complete the demonstration: $tot\_elves_k = OPT - uncovered\_elves_k \geq OPT - \frac{OPT}{e} = (1 - \frac{1}{e}) \cdot OPT \Rightarrow SOL \geq (1 - \frac{1}{e}) \cdot OPT$.
So, **it is possible to conclude that the greedy algorithm has, for large numbers of *k*,** $(1 - \frac{1}{e})$ **approximation factor**.

2 Exercise 2

2.a) The formulation as an ILP problem and its relaxation

The problem proposed can be modeled as a graph, in which there are n destinations for n goods to be shipped, each good has three possible paths to follow and these paths are routes, sequences of road segments, representing the edges of the graph. These three paths for each good are offered by three different cargo companies and the goal is to choose just one of them for each good to be transported. Since there are frequent road blockages, in the rural area represented by the graph, the objective is that not too many of the transports are obstructed. In other words, the aim is to **pick the paths** $P_{i,j}$ for each good g_i in such a way that **the maximum number of times that an edge is crossed by the paths chosen for the goods is minimized**. In order to solve it, a possible way is to **model the problem as an ILP** (Integer Linear Programming) problem, which is a problem that expresses the optimization of a linear function subject to a set of linear constraints using integer variables, and then **relax it to an according LP** (Linear Programming) problem, computing a feasible solution starting from the LP optimum giving a **rounding algorithm that approximates the optimal solution**. It is possible to define, by using the problem's data, an ILP problem as follows: the idea is to assign for each edge of the 'roads network' a value which represents the number of chosen trucks that cross this edge. So, the next step is to define the value a which has to be the maximum of these values and the problem wants to minimize a. The best way to do it is to use the **min-max model** to transform non-linear objective functions into linear ones: it just consists in creating the variable a and, since it represents the maximum of the values related to the edges (the number of times each of them is crossed by a truck chosen), the trick is to set a constraint for each value expressing the condition that it has to be greater than each of them to be their maximum and, being equal to the maximum value, it will be the minimum value that is greater than each of them, therefore the objective function will be min(a). The value a, for each edge, is defined as the sum for each good and for each company truck of the product between two values: the first one is $P_{i,j}$, its value is *1* if and only if for the i^{th} good it was chosen the j^{th} company, it is *0* otherwise; the second one is $X_{i,j,e}$ and its value is *1* if the edge e is contained in the path for the i^{th} good offered by the j^{th} company. It is also required to define the constraint according to which each good must be transported by only one truck. Summing all these requirements, **the ILP formulation (on the left)** is the following:

$$\begin{aligned}&\min\ a\\ &\textstyle\sum_{i=i}^{n}\sum_{j=i}^{3} P_{i,j}\cdot X_{i,j,e}\le a \quad ,\forall e\in E\\ &P_{i,1}+P_{i,2}+P_{i,3}=1 \quad ,\forall i\in[1,...,n]\\ &P_{i,j}\in\{0,1\}\\ &X_{i,j,e}=\begin{cases}1 & \textit{if } e\in P_{i,j}\\ 0 & \textit{otherwise}\end{cases}\end{aligned}$$

$$\begin{aligned}&\min\ a\\ &\textstyle\sum_{i=i}^{n}\sum_{j=i}^{3} P_{i,j}\cdot X_{i,j,e}\le a \quad ,\forall e\in E\\ &P_{i,1}+P_{i,2}+P_{i,3}=1 \quad ,\forall i\in[1,...,n]\\ &P_{i,j}\in[0,1]\\ &X_{i,j,e}=\begin{cases}1 & \textit{if } e\in P_{i,j}\\ 0 & \textit{otherwise}\end{cases}\end{aligned}$$

The LP problem (on the right) can be constructed as a relaxation of the ILP one and it is the problem in which the integrality constraints of each variable are removed. **The relaxation technique is used to transform an NP-hard optimization problem (ILP) into a related problem solvable in polynomial time (LP).**

2.b) The rounding algorithm and the 3-approximation guarantee

The next step is to define a rounding algorithm, to compute a feasible solution of the ILP problem from the solution given by the LP problem. **As the set of solutions of the ILP problem is a subset of that of the LP problem, considering a^* as the value of the objective function given by the solutions of the LP problem and OPT as the optimal solution of the ILP problem: $a^* \le$ OPT** (1). Therefore, **the rounding algorithm goes from a fractional solution of the problem, obtained through the LP problem, to an integer solution**. The idea is to use a deterministic method (**deterministic rounding algorithm**) to convert an optimal solution of the relaxed problem into an approximately optimal solution to the original problem. Using the rounding algorithm, the goal is to round the values $P^*_{i,j}$, the solution of the LP problem, so that it is possible to obtain a valid solution for the ILP problem. In order to round the values, it is needed to define, for each good, three variables (one for each trucks' company) $\overline{P_{i,j}}$, which can assume *1* or *0*: $\overline{P_{i,j}}$ assumes *1* if the correspondent $P^*_{i,j}$ is the higher value for the i^{th} good, otherwise it assumes *0*.

Algorithm 2 Rounding

```
Require: g ⇐ set of goods, $P^*_{i,j}{}'s$ ⇐ solutions' values given by the LP problem.

  procedure Ex_2(g, $P^*_{i,j}{}'s$):
      for each good i do
          The values $P^*_{i,j}{}'s$ are inserted into a list in a decreasing way.
          It takes the first value $P^*_{i,j^1}$ and it assigns the corresponding $\overline{P_{i,j^1}}$=1 and it assigns $\overline{P_{i,j^2}}$=0 and $\overline{P_{i,j^3}}$=0.
      end for
```

If the three values of $P^*_{i,j}{}'s$ have the same value $\frac{1}{3}$, then *1* is randomly assigned to only one of the three, so **a feasible solution is achieved in any case**, and, while rounding, the value of a path can increase at most by a factor of 3: $\mathbf{\overline{P_{i,j}} \le 3 \cdot P^*_{i,j}}$ (2). Therefore the optimal value (OPT) of a will be caused by some paths, and, if the value of each of these paths increases by a factor of n through the rounding, it will be a **3-approximation**. Mathematically, considering $\overline{a}$ as the value of the objective function given by the rounded integer solutions and using, in order, (2) and (1): $\mathbf{\overline{a} \le 3\cdot\ a^* \le 3 \cdot OPT}$.

3 Exercise 3

Brunetti, Carmignani

3.a) The global minimum cut problem and its variation

This exercise provides a variation of the famous global min-cut problem, both of them are **randomized algorithms**, based on random choices. In order to reduce the probability of failure, the objective is to use a technique known as "parallel amplification", consisting in different parallel repetitions of the same algorithm in order to get the best result among them. Before starting to analyze the variation, it is important to review the randomized algorithm for global min-cut. Given an undirected and connected graph G=(V, E) of n nodes, the goal is to find a cut (A, B) of minimum cardinality. The best known solution can be achieved using a randomized algorithm, called "**the contraction algorithm**". This algorithm is based on a cycle and many operations of contraction: take an edge e=(u, v) at random, contract the edge e, substitute u and v with a unique super-node w and repeate it until the graph has only 2 nodes u_1 and v_1. The contraction operation consists in substituting u and v with a single node w, a supernode, preserving the edges updating the endpoints of u and v to w, but deleting the self-loops. At the end, the algorithm returns the cut as all nodes which have been contracted creating the node v_1. The study of this algorithm led to three important **statements**:

1. **The probability that the algorithm contracts an edge in the min-cut F^* in an iteration i is $\leq \frac{\mathbf{2}}{\mathbf{n-(i-1)}}$.**
2. **The probability that the algorithm has success in each iteration, and so it returns a min-cut, is $\geq \frac{\mathbf{2}}{\mathbf{n(n-1)}}$.**
3. **Using "parallel amplification", running the algorithm $n^2 \cdot ln(n)$ times, gives a probability of failure $\leq \frac{\mathbf{1}}{\mathbf{n^2}}$.**

Starting from these statements, it is possible to evaluate the **variation of the algorithm** proposed: from the graph G with n nodes, use the randomized algorithm contracting the graph until a graph G_k is reached with $k=\sqrt{n}$ nodes, then create the $l = \sqrt{n}$ copies of G_k and make l executions of the algorithm; the minimum value returned among them will be the final output.

3.a) The probability that the reduced graph has the same minimum cut-set value of the original graph

The reduced graph G_k has the same min cut-set value of G if no edge in the min cut-set F^* is contracted in the first $n' = n - \sqrt{n}$ iterations. Consider E_j as the event "the edge contracted in iteration j is not in F^*" and Z_j as the event "no edge in F^* is contracted in the first j iterations". Then, using conditional probability, the **statement 1** and the probability of a complementary event: $\mathbb{P}(Z_{n'}) = \mathbb{P}(\bigcap_{j=1}^{n'} E_j) = \mathbb{P}(E_1) \cdot \mathbb{P}(E_2|E_1) \cdot ... \cdot \mathbb{P}(E_{n'}|E_1 \cdot E_2 \cdot ... \cdot E_{n'-1}) \geq \prod_{i=1}^{n'}(1 - \frac{2}{n-(i-1)})$. Just computing these factors from 1 to n' and cancelling out parts of them, it is trivial to see that: $\mathbb{P}(Z_{n'}) \geq \frac{\sqrt{n}\cdot(\sqrt{n}-1)}{n\cdot(n-1)} = \frac{1-\frac{\sqrt{n}}{n}}{n-1} \geq \frac{1-\frac{\sqrt{n}}{n}}{n}$. By induction, it is possible to prove that $\frac{1-\frac{\sqrt{n}}{n}}{n} \geq \frac{1}{2n}$, $\forall n \geq 4$. The case of $n = 2$ is trivial since no contractions are performed, whereas $n = 3$ is the case in which at most one contraction is done so the probability is at least $(1 - \frac{2}{n}) = \frac{1}{3} = \frac{1}{n}$. Therefore, it holds for each possible value of n: $\mathbb{P}(\mathbf{Z_{n'}}) \geq \frac{\mathbf{1}}{\mathbf{2n}}$.

3.b) The probability that the algorithm outputs a correct min-cut set

The probability that the variation of the algorithm proposed gives a correct min cut-set can be evaluated using the **statement 2**, in fact, it outputs a correct min-cut set if at least one of the copies is correct. Now, let's consider $Z_{n'}$ as the event "the reduced graph G_k has the same min-cut value of the original graph", and N as the event "at least one of the $\sqrt{n}$ copies of G_k leads to a correct min-cut set". Then, X is the event "the algorithm outputs a correct min-cut set", and $X = Z_{n'} \bigcap N$. By independence of events, it is true that: $\mathbb{P}(X) = \mathbb{P}(Z_{n'} \bigcap N) = \mathbb{P}(Z_{n'}) \cdot \mathbb{P}(N)$. From 3.a), $\mathbb{P}(Z_{n'}) \geq \frac{1}{2n}$, so $\mathbb{P}(X) \geq \frac{1}{2n} \cdot \mathbb{P}(N)$. Let's consider $\mathbb{P}(N)$, because of the probability of complementary events, $\mathbb{P}(N) = 1 - \mathbb{P}(\overline{N})$. The complementary event of N, $\overline{N}$, is the event "all of the $\sqrt{n}$ copies will fail", meaning that none of them will retrieve the min-cut set of G_k. Given the independence of each execution from the others on the reduced graph, since the reduced graph G_k has $\sqrt{n}$ nodes, the **statement 2**, valid when running the algorithm on a graph of n nodes, becomes such that the probability that the algorithm running on a copy of G_k has success is $\geq \frac{2}{\sqrt{n}(\sqrt{n}-1)}$. Therefore, the probability that a copy fails is $\leq 1 - \frac{2}{\sqrt{n}(\sqrt{n}-1)}$. The event $\overline{N}$ is defined as "all the $\sqrt{n}$ copies will fail", so, by independence of events: $\mathbb{P}(\overline{N}) \leq (1 - \frac{2}{\sqrt{n}(\sqrt{n}-1)})^{\sqrt{n}}$. For $a << b$, $(1 - \frac{1}{b})^a \leq 1 - \frac{a}{b}$, so just defining a= $\sqrt{n}$ and b= $\frac{\sqrt{n}(\sqrt{n}-1)}{2}$, it follows that: $\mathbb{P}(\overline{N}) \leq (1 - \frac{2}{\sqrt{n}(\sqrt{n}-1)})^{\sqrt{n}} \leq 1 - \frac{2}{\sqrt{n}-1} \leq 1 - \frac{1}{\sqrt{n}}$. Now, it is possible to compute $\mathbb{P}(N) = 1 - \mathbb{P}(\overline{N}) \geq \frac{1}{\sqrt{n}}$, and, consequently, it leads to: $\mathbb{P}(\mathbf{X}) \geq \frac{\mathbf{1}}{\mathbf{2n}} \cdot \frac{\mathbf{1}}{\sqrt{\mathbf{n}}} = \frac{\mathbf{1}}{\mathbf{2n^{3/2}}}$.

3.c) Comparing contractions and resulting error probabilities to that when running the original algorithm twice

Since a contraction is done in each step until 2 nodes remain, the number of contractions of the original algorithm is trivially n-2, so, if it is executed twice, it will be $2 \cdot (n-2)$. Similarly, the proposed variation has a number of contractions equal to $(n-\sqrt{n}) + \sqrt{n} \cdot (\sqrt{n} - 2) = 2n - 3\sqrt{n}$. More generally, **both the solutions have the same order of number of contractions: $\mathbf{O(n)}$**. Let's compare the error probability: for 3.b), the proposed variation has a success probability $\geq \frac{1}{2n^{3/2}}$, which is $\mathbf{\Omega(n^{-\frac{3}{2}})}$, the original algorithm executed twice, because of **statement 2**, has success, being this event the complementary of the event "neither of the executions have success", with probability $\geq 1 - (1 - \frac{2}{n(n-1)})^2$, which is asymptotically $\mathbf{\Omega(n^{-2})}$. Therefore, **it follows that the upper bound of the error probability of the original algorithm executed twice is greater than the upper bound of the error probability of the variation**. As it was thought, similarly to **statement 3** and to the Karger-Stein algorithm, the repetition of the original algorithm just two times independently from the number of nodes is not a good solution and the order of the probability of success is the same of running it just one time, instead, running the variation of the algorithm provides a good improvement, since early iterations are less risky and then, when the number of nodes is $\sqrt{n}$, it uses "parallel amplification" creating a number of copies which depend on the number of nodes.

4 Exercise 4

Brunetti, Carmignani

4.a) The problem in the computational game theory

This problem can be modeled as a usual problem in **the computational game theory**. The **players** are represented by the **reviewers**, the **strategies** are defined as the **sets of the papers reviewed by a reviewer**, whereas the **utility** of each player is defined as the **difference between the rewards, equally divided among the reviewers, obtained from the papers that he reviews and their costs ($s_{i,j}$)**, which cannot exceed the deadline (T). The **assumption** to be respected is that there has to be a combination of strategies such that **every reviewer has positive utility and all papers get at least one review**. The **goal** is to find the **fraction of papers that receive at least 1 review at a pure Nash equilibrium, satisfying the assumption**.

4.a) The set of costs such that the fraction of paper is close to $\frac{1}{n}$ having the reward B=1

In order to find a set of $s'_{i,j}s$ such that the number of the papers that receive the review is $\frac{1}{n}$, it is necessary to define a generic instance of problem that induces the reviewers **to review only one paper**. Each reviewer must choose a strategy which consists in reviewing only a subset of the papers in order to maximize his utility. Therefore, it is about to define the concept of **pure Nash equilibrium** (PNE): this condition is reached when **no player can do better by unilaterally changing his strategy**. So, it is necessary to define a strategy through which each reviewer never changes his strategy , assuming that no one changes his strategy. **The idea is to divide the papers into several groups according to their costs**. In order to show that for a set of $s_{i,j}$'s the number of papers that receive reviews is close to $\frac{1}{n} \cdot n$, it is possible to define a set of costs that drives **all the reviewers to review the same single paper**:

$$\begin{cases} paper_1 \Rightarrow \textit{1 paper costs } \varepsilon & \textit{with}: 0 < \varepsilon << 1, \varepsilon < \frac{1}{k}, \varepsilon \leq T, \varepsilon < \frac{1}{k \cdot n} \\ papers_{nr} \Rightarrow \textit{group of (n - 1) papers, which cost } (1 + \varepsilon) \end{cases}$$

It must respect **several conditions to make sure that the constraints are respected**. At first ε must be smaller than the reward obtained by the *paper*$_1$ because otherwise it would be incovenient to review the *paper*$_1$. Another condition about ε is that this value must be less or equal than the deadline T, otherwise no one could review *paper*$_1$ without violating the constraints. Looking at the defined costs, it can be said that, for this problem formulation, PNE is obtained when all the reviewers choose to review only the paper of the first group. It is possible to understand this statement through the following table:

B=1	All the reviewers review only *paper*$_1$. Utility for each i^{th} reviewer $\rightarrow$ u$_i = \frac{1}{k} - \varepsilon, u_i > 0$	The reviewer x reviews *paper*$_1$ and *paper*$_2$ $\in$ *papers*$_{nr}$. Each reviewer i ($i \neq x$) keeps unchanged his strategy. $u_x = \frac{1}{k} - \varepsilon + 1 - 1 - \varepsilon$, it makes utility worse.

In the table, the first column shows the value of the reward for each paper, the second column shows a PNE for the defined costs, the third column shows how **a reviewer, that moves from the Nash equilibrium condition, makes worse his utility**, like also when he decides to not review any paper. Now, it is needed to **show that the asssumption is true**. Supposing that the reviewer x reviews all the papers and the other reviewers review only *paper*$_1$. This condition verify the statement according to which all the papers get at least one review and the conditions on ε verify the assumption by which each reviewer has positive utility. In fact, it is true for the reviewers that review only *paper*$_1$ and it is true also for x, this because: $u_x = \frac{1}{k} - \varepsilon + (1 - 1 - \varepsilon) \cdot (n-1) = \frac{1}{k} - \varepsilon \cdot n$, it is always true by the ε condition by which $\varepsilon < \frac{\mathbf{1}}{\mathbf{k \cdot n}}$.

4.b) Increasing the reward to B=2 the fraction of papers is close to $\frac{1}{3}$

Through the statements above, it is possible to **generalize the problem depending on the reward value B**, so that, increasing B, it increases the number of the reviewed papers. It is possible to divide the papers into three groups, each one with a certain cost. It is possible to define them as follows:

$$\begin{cases} paper_1 \Rightarrow \textit{1 paper costs } \varepsilon & \textit{with}: 0 < \varepsilon << 1, \varepsilon < \frac{1}{k}, \varepsilon \leq T, \varepsilon < \frac{1}{k \cdot \frac{2 \cdot n}{B+1}} \\ papers_r \Rightarrow \textit{group of } \frac{B-1}{B+1} \cdot n \textit{ papers which cost } \varepsilon \\ papers_{nr} \Rightarrow \textit{group of } (\frac{2}{B+1} \cdot n) - 1 \textit{ papers which cost } (B + \varepsilon) \end{cases}$$

By setting B = 2, it gives that *paper*$_1$ has cost ε, each paper of *papers*$_r$ has cost ε and each paper of *papers*$_{nr}$ has cost $(2 + \varepsilon)$. Through the following table, it is possible to understand how the reviewers work when the reward is equal to 2:

B=2	Each reviewer x reviews *paper*$_1$ and all papers in *papers*$_r$. $u_x = \frac{1}{k} - \varepsilon + (\frac{2}{k} - \varepsilon) \cdot \frac{1}{3}$, it improves the utility.	The reviewer x reviews *paper*$_1$ and all papers in *papers*$_r$ and one paper in *papers*$_{nr}$. Each reviewer i ($i \neq x$) keeps unchanged his strategy. $u_x = \frac{1}{k} - \varepsilon + (\frac{2}{k} - \varepsilon) \cdot \frac{1}{3} + (2 - 2 - \varepsilon)$, it makes utility worse.

In the table, the first column shows the value of the reward for each paper, the second column shows **a PNE for the defined costs** and the third column shows how a reviewer, that moves from the Nash equilibrium condition, makes worse his utility. Another way to move from Nash equilibrium is when the x reviewer decides to not review all the papers of the second group, in this case he makes worse his utility because he looses the positive utility given by that papers. So, **when B=2, the fraction of papers reviewed at a PNE is close to $\frac{1}{3}$**, since the first paper is reviewed and also the group *paper*$_r$ which is equal to $\frac{\mathbf{1}}{\mathbf{3}}$.