The exam paper in the first position of a partition is assigned 1 point, the second position assigns 2 points, the third position assigns 3 points, and so on. This creates \( n \) sets of \( k + 1 \) elements; every set defines a student and the \( k \) exam papers she has to grade.

**Computation of a perfect matching:**

- For each node \( u \in U \), select uniformly at random an edge among its incident nodes, remove it and continue for the remaining nodes;
- If \( u \) does not have any incident edges, then restart the matching computation from scratch.

**Properties:**

- Every exam paper must be contained in exactly \( k \) bundles;
- Every student must be assigned a bundle not containing her own exam paper.

**The Algorithm:**

- Select \( k + 1 \) pairwise disjoint perfect matchings on the complete bipartite graph \( G_{n,n} = (U, V, E) \);
- This creates \( n \) sets of \( k + 1 \) elements;
- Every set defines a student and the \( k \) exam papers she has to grade.

**Aggregation Methods**

- **co-rank** supports three aggregation methods motivated by social choice theory; the function of all of them is similar.
- Every exam paper is assigned points according to the position it has in the partial rankings it participates.
- The score of an exam paper equals the total number of its ordinal ranks.
- Ties are resolved randomly.

**Borda count:**

- It is characterized by the scoring vector \( (k, k - 1, \ldots, 1) \).
- The exam paper in the first position of a partial ranking is assigned \( k \) points, the exam paper in the second position is assigned \( k - 1 \) points, and so on.

**Partition:**

- Every student approves exactly \( k/2 \) exam papers.
- Only the exam papers that have been approved are assigned a point each.

**Randomized approval:**

- Every student approves a random number of exam papers, selected uniformly at random from the set \( \{1, \ldots, k - 1\} \).
- Again, only the exam papers that have been approved are assigned a point each.

**Technologies used**

- CSS, HTML, JS, Bootstrap, PHP, MySQL.

**Functionalities – System workflow**

**Initialization phase:**

- The instructor creates a new exam;
- She uploads a file containing the exam questions;
- She defines the submission and grading deadlines;
- She defines communication rules among students and instructor – (uni/bi)directional;
- She defines the aggregation method.

**Examination phase:**

- Every student downloads the exam question;
- She answers them and uploads a file containing her answers.

**Grading phase (after the submission deadline):**

- The instructor manually initiates the grading process;
- A bundle computation algorithm computes a set of bundles of exam papers;
- Every student is assigned a bundle (not containing her own exam paper);
- Through the user-friendly interface of co-rank tool, each student orders the exam papers in her bundle;
- After the grading deadline, the instructor initiates the aggregation of the partial rankings;
- A global ranking of the students is computed and announced.

**Bundle computation**

- Every exam paper must be contained in exactly \( k \) bundles;
- Every student must be assigned a bundle not containing her own exam paper.

**The Algorithm:**

- Select \( k + 1 \) pairwise disjoint perfect matchings on the complete bipartite graph \( G_{n,n} = (U, V, E) \);
- This creates \( n \) sets of \( k + 1 \) elements;
- Every set defines a student and the \( k \) exam papers she has to grade.

**Demo**

- Create a new account with the role of instructor and a new account with the role of a student.
- Create a new exam with 10,000 students.
- Present the functionalities provided by the co-rank tool in both cases through a step-by-step scenario.

**co-rank website**

The co-rank tool can be accessed through the url:

co-rank.ceid.upatras.gr

**More info (theory)**

Ioannis Caragiannis, George A. Krimpas, and Alexandros A. Voudouris, Aggregating partial rankings with applications to peer grading in massive online open courses. AAMAS 2015, pp. 675–683.