Today:

- Overview of course.
- Overview of peer grading.
This Class

• paper reading
  (roughly three per week)

• student presentations
  (with practice presentation)

• student projects (theoretical or empirical, with data from Northwestern classes)
This Class

- paper reading
  (roughly three per week)

- student presentations
  (with practice presentation)

- student projects (theoretical or empirical, with data from Northwestern classes)
  - proposal (week 4)
  - literature review (week 6)
  - first draft (week 9)
  - in class presentation (week 10)
  - final draft (exam week, a.k.a., 11)
Week 0: Introductory lecture on peer grading (today; no readings)

Week 1: Peer grading systems (general)
Week 2: Peer prediction (game theory, human computation)
Week 3: Eliciting peer feedback (HCI, learning science)
Week 4: Incentivizing effort and accuracy (scoring rules, auctions)
Week 5: Assigning reviews (algorithms, human computation)
Week 6: Cardinal grade aggregation (machine learning, algorithms)
Week 7: Accuracy of peer reviews (HCI, learning science)
Week 8: Ordinal grade aggregation (game theory, machine learning)
Week 9: Evaluating learning outcomes (learning science)

Week 10: Project presentations (no readings)
Data for Projects

**Data Set 1:** Computer Science for Everyone (EECS 101)

- two assignments (mini-essays) per week.
- 250 students.
- three peer reviews per student per essay.
- detailed specific rubrics.
- TA reviews for 40 submissions per assignment
Data for Projects

Data Set 1: Computer Science for Everyone (EECS 101)
- two assignments (mini-essays) per week.
- 250 students.
- three peer reviews per student per essay.
- detailed specific rubrics.
- TA reviews for 40 submissions per assignment

Data Set 2: Introduction to Algorithms (EECS 336)
- two assignments (problems) per week.
- 90 students (submissions in pairs)
- three peer reviews per student per problem.
- detailed specific rubrics.
- TA reviews for 10 submissions per assignment.
**Philosophy:** The next frontier for computer science is understanding and designing computation outside digital computers.
Philosophy: The next frontier for computer science is understanding and designing computation outside digital computers.

Question: What can computer science say about teaching a course?
Computer Science on Teaching

**Philosophy:** The next frontier for computer science is understanding and designing computation outside digital computers.

**Question:** What can computer science say about teaching a course?

**Computational Model:**

- Students: strategic agents
- TAs/Instructor: (noisy) computers
- Syllabus: maps histories of actions to a grade in the class.
- Student Incentives: minimize work, maximize grade.
- Objective: minimize work, maximize learning, fairly assess.
Philosophy: The next frontier for computer science is understanding and designing computation outside digital computers.

Question: What can computer science say about teaching a course?

Computational Model:
- Students: strategic agents
- TAs/Instructor: (noisy) computers
- Syllabus: maps histories of actions to a grade in the class.
- Student Incentives: minimize work, maximize grade.
- Objective: minimize work, maximize learning, fairly assess.

Interdiciplinarity: must combine
- computational models (e.g., algorithms, machine learning, human computer interaction),
- economic models (e.g., game theory, auctions),
- learning science models (e.g., scaffolding, learning outcomes, interventions).
Advantages of Peer Grading:

- learning by reviewing.
- reduces teacher grading.
- promptness of feedback.
- enables data mining.

Potential Disadvantages: Inaccurate grades, student unrest, ...
Advantages of Peer Grading:

Advantages of Peer Grading: (observations from Intro to Algs)

- learning by reviewing.
- reduces teacher grading.
- promptness of feedback.
- enables data mining.

Potential Disadvantages: Inaccurate grades, student unrest, ...
Advantages of Peer Grading: (observations from Intro to Algs)

- learning by reviewing.
  (learn material: 60% agree; learn to write better: 55% agree)
  (worse students agree more: A: 52%; B: 54%; C: 75%; D: 80%)

- reduces teacher grading.

- promptness of feedback.

- enables data mining.

Potential Disadvantages: Inaccurate grades, student unrest, ...
Advantages of Peer Grading: (observations from Intro to Algs)

- learning by reviewing.
  (learn material: 60% agree; learn to write better: 55% agree)
  (worse students agree more: A: 52%; B: 54%; C: 75%; D: 80%)

- reduces teacher grading.
  (TAs graded 1/5 of student work.)

- promptness of feedback.

- enables data mining.

Potential Disadvantages: Inaccurate grades, student unrest, . . .
Advantages of Peer Grading: (observations from Intro to Algs)

• learning by reviewing.
  (learn material: 60% agree; learn to write better: 55% agree)
  (worse students agree more: A: 52%; B: 54%; C: 75%; D: 80%)

• reduces teacher grading.
  (TAs graded 1/5 of student work.)

• promptness of feedback.
  (peer review feedback in 3 days, grades in 5 days; versus 2 weeks)

• enables data mining.

Potential Disadvantages: Inaccurate grades, student unrest, …
Advantages of Peer Grading: (observations from Intro to Algs)

- learning by reviewing.
  (learn material: 60% agree; learn to write better: 55% agree)
  (worse students agree more: A: 52%; B: 54%; C: 75%; D: 80%)

- reduces teacher grading.
  (TAs graded 1/5 of student work.)

- promptness of feedback.
  (peer review feedback in 3 days, grades in 5 days; versus 2 weeks)

- enables data mining.
  (50 submissions × 18 problems × 6 peer reviews × 8 rubric elements = 43200 scores)

Potential Disadvantages: Inaccurate grades, student unrest, ...
Advantages of Peer Grading: (observations from Intro to Algs)

- learning by reviewing.
  (learn material: 60% agree; learn to write better: 55% agree)
  (worse students agree more: A: 52%; B: 54%; C: 75%; D: 80%)

- reduces teacher grading.
  (TAs graded 1/5 of student work.)

- promptness of feedback.
  (peer review feedback in 3 days, grades in 5 days; versus 2 weeks)

- enables data mining.
  (50 submissions × 18 problems × 6 peer reviews × 8 rubric elements = 43200 scores)

Potential Disadvantages: Inaccurate grades, student unrest, . . .
(3.7% appeal rate; 1-6% strongly disagree with survey questions)
Peer Grading Systems

System Components: [Week 1]

- user interface [Week 3]
- backend data management
- peer grading algorithms
Peer Grading Systems

System Components: [Week 1]

• user interface [Week 3]

• backend data management

• peer grading algorithms

Main Algorithms:

• matching algorithm (who grades what)

• submission grading algorithm (from peer and TA reviews)

• review grading algorithm (from peer and TA reviews)
Peer Grading Systems

System Components: [Week 1]

- user interface [Week 3]
- backend data management
- peer grading algorithms

Main Algorithms:

- matching algorithm (who grades what)
- submission grading algorithm (from peer and TA reviews)
- review grading algorithm (from peer and TA reviews)

Agenda: summarize algorithms; connect to course topics.
Submission Grading Algorithms:
compute grades for submissions from peer and TA reviews
Submission Grading Algorithm

Submission Grading Algorithms:
compute grades for submissions from peer and TA reviews

- E.g., via the expectation maximization algorithm
- peer accuracy (variance),
  submission grade (expectation) and clarity (variance).
Submission Grading Algorithms:
compute grades for submissions from peer and TA reviews

- E.g., via the expectation maximization algorithm
- peer accuracy (variance), submission grade (expectation) and clarity (variance).

Course Topics:

- Cardinal grade aggregation (machine learning) [Week 6]
- Accuracy of peer reviews (HCI, learning science) [Week 7]
- Ordinal grade aggregation (algorithms, machine learning) [Week 8]
Matching Algorithms:
choose peer and TA matching in advance of reviews.
Matching Algorithms:
choose peer and TA matching in advance of reviews.

- minimize number of TA reviews
- maximize quality of grades from peer reviews.
Matching Algorithms:
choose peer and TA matching in advance of reviews.

- minimize number of TA reviews
- maximize quality of grades from peer reviews.
- E.g.: $n$ peers, $m$ submissions, $k$ reviews per peer, $\ell$ TA reviews.
Matching Algorithms:
choose peer and TA matching in advance of reviews.

- minimize number of TA reviews
- maximize quality of grades from peer reviews.
- E.g.: \( n \) peers, \( m \) submissions, \( k \) reviews per peer, \( \ell \) TA reviews.
  - assign TA to \( \ell \) random submissions.
  - uniform random 1-to-many match peers to these submissions.
  - uniform random \((k - 1)\)-to-many match peers to remaining.
Matching Algorithms:
choose peer and TA matching in advance of reviews.

- minimize number of TA reviews
- maximize quality of grades from peer reviews.
- E.g.: $n$ peers, $m$ submissions, $k$ reviews per peer, $\ell$ TA reviews.
  - assign TA to $\ell$ random submissions.
  - uniform random 1-to-many match peers to these submissions.
  - uniform random $(k - 1)$-to-many match peers to remaining.
- Intro to Algs: $n \approx 90; m \approx 50; k = 3; \ell = 10$. 
Matching Algorithms:
choose peer and TA matching in advance of reviews.

- minimize number of TA reviews
- maximize quality of grades from peer reviews.
- E.g.: \( n \) peers, \( m \) submissions, \( k \) reviews per peer, \( \ell \) TA reviews.
  - assign TA to \( \ell \) random submissions.
  - uniform random 1-to-many match peers to these submissions.
  - uniform random \((k - 1)\)-to-many match peers to remaining.
  - Intro to Algs: \( n \approx 90; m \approx 50; k = 3; \ell = 10 \).

Course Topics:
- Assigning reviews (algorithms, human computation) [Week 5]
Review Grading Algorithm:
compute grades for peer reviews from peer and TA reviews
Review Grading Algorithm:
compute grades for peer reviews from peer and TA reviews

- incentive issues:
  - accuracy
  - effort
Review Grading Algorithm:
compute grades for peer reviews from peer and TA reviews

- incentive issues:
  - accuracy
  - effort

Course Topics:

- Peer prediction *(game theory, human computation)* [Week 2]
- Eliciting peer feedback *(HCI, learning science)* [Week 3]
- Incentivizing effort and accuracy *(scoring rules, auction design)* [Week 4]
Review Grading Algorithm:
compute grades for peer reviews from peer and TA reviews

- incentive issues:
  - accuracy
  - effort

Course Topics:

- Peer prediction (game theory, human computation) [Week 2]
- Eliciting peer feedback (HCI, learning science) [Week 3]
- Incentivizing effort and accuracy (scoring rules, auction design) [Week 4]

Next: accuracy via proper scoring rules; effort via all-pay auctions
Incentivizing Accurate Reviews

From other peer reviews: [Week 2]
Incentivizing Accurate Reviews

From other peer reviews: [Week 2]

From TA reviews: [Week 4]

• idea: cf. proper scoring rules
Incentivizing Accurate Reviews

From other peer reviews: [Week 2]

From TA reviews: [Week 4]

• idea: cf. proper scoring rules

• e.g., quadratic: \[ \text{review-grade} = 1 - (\text{ta-score} - \text{peer-score})^2 \]
Incentivizing Accurate Reviews

From other peer reviews: [Week 2]

From TA reviews: [Week 4]

• idea: cf. proper scoring rules

• e.g., quadratic: \( \text{review-grade} = 1 - (\text{ta-score} - \text{peer-score})^2 \)

• issue: “good for incentives”, inaccurate for assessment of learning.
  (proper scoring rules are convex)
Incentivizing Effort in Reviews
Idea: model as all-pay auctions
Incentivizing Effort in Reviews

**Idea:** model as all-pay auctions

- *linear model:*
  - utility = grade − effort

- cf. *all-pay auctions:*
  - utility = value × alloc − payment
  - maximizing revenue = “maximizing accuracy”
**Idea:** model as all-pay auctions

- *linear model:*
  - utility = grade − effort
  - accuracy = effort × skill

- cf. *all-pay auctions:*
  - utility = value × alloc − payment
  - maximizing revenue = “maximizing accuracy”
Idea: model as all-pay auctions

- **linear model:**
  - utility = grade − effort
  - accuracy = effort × skill
  - skill × utility = skill × grade − accuracy

- cf. **all-pay auctions:**
  - utility = value × alloc − payment
  - maximizing revenue = “maximizing accuracy”
Idea: model as all-pay auctions

- linear model:
  - utility = grade − effort
  - accuracy = effort × skill
  - skill × utility = skill × grade − accuracy

- cf. all-pay auctions:
  - utility = value × alloc − payment
  - maximizing revenue = “maximizing accuracy”
Week 0: Introductory lecture on peer grading (today; no readings)

Week 1: Peer grading systems (general)
Week 2: Peer prediction (game theory, human computation)
Week 3: Eliciting peer feedback (HCI, learning science)
Week 4: Incentivizing effort and accuracy (scoring rules, auctions)
Week 5: Assigning reviews (algorithms, human computation)
Week 6: Cardinal grade aggregation (machine learning, algorithms)
Week 7: Accuracy of peer reviews (HCI, learning science)
Week 8: Ordinal grade aggregation (game theory, machine learning)
Week 9: Evaluating learning outcomes (learning science)

Week 10: Project presentations (no readings)