אוניברסיטת
אריאל
בשומרון

Final project

Optimization-based Mechanisms for the Course Allocation Problem

Moriya Ester Ohayon
Tamar Bar-Ilan
Ofek Kats

project advisor: Prof. Erel Segal-Halevi

Based on "Optimization-based Mechanisms for the Course Allocation Problem",
by Hoda Atef Yekta, Robert Day (2020, https://doi.org/10.1287/ijoc.2018.0849)

Course Allocation Problem (CAP)

In the course allocation problem, a university administrator seeks to efficiently and fairly allocate seats (or items) in over-demanded courses among students (or agents) with heterogeneous preferences.

Fairpyx- Python library containing various algorithms for fair allocation, with an emphasis on Course allocation. Maintained by Prof. Erel Segal-Halevi.

Algorithms:

- **TTC (Top Trading-Cycle)**
- **SP (Second Price)**
- **TTC-O**
- **SP-O**
- **OC (Ordinal-then-Cardinal)**

GitHub Code: https://github.com/Final-Project-fairpyx/fairpyx/

Demo website: https://tome.csariel.xyz/

Project steps

- Summary of the Article
- Created Manual Examples
- Conducted Code Tests
- Fully Implemented the Algorithms (TTC, SP, TTC-O, SP-O, OC)
- Compared Performance
- Enhanced Performance
- Submitted Pull Request – to Fairpyx library
- Created Demo Website
- Project Book

TTC - Top Trading-Cycle

The Top Trading Cycles (TTC) algorithm allocates courses to students based on their bids.

Each student bids on all courses with a budget of 1000 points and "votes" for the course they bid the most on.

Courses are allocated to the top bidders who voted for that course. After allocation, the voted course is removed from the student's options, and if a course is received, overlapping and full courses are also removed.

If a student doesn't receive their chosen course, the algorithm runs additional rounds for these students until all receive a course or have no courses left.

The process then repeats with all students.

TTC - Top Trading-Cycle

The Top Trading Cycles (TTC) algorithm allocates courses to students based on their bids.

Each student bids on all courses with a budget of 1000 points and "votes" for the course they bid the most on.

Courses are allocated to the top bidders who voted for that course. After allocation, the voted course is removed from the student's options, and if a course is received, overlapping and full courses are also removed.

If a student doesn't receive their chosen course, the algorithm runs additional rounds for these students until all receive a course or have no courses left.

The process then repeats with all students.

SP - Second Price

This algorithm incorporates a point refund mechanism.

In each round, students vote for the top course on their available list, and courses are allocated to the highest bidders. If a course has enough space, it's deemed "free," and students' bid points are transferred to their next choice.

If space is limited, the course price is set to the highest bid from rejected students. Successful students pay this price, with remaining points transferred to their next choice, while rejected students keep all their points for the next course.

The algorithm repeats rounds for rejected students until all receive a course or have no options left, then starts a new phase with all students.

אוניברסיטת
אריאל
בשומרון

TTC-O - Top Trading-Cycle Optimization

The TTC-O algorithm operates similarly to the standard TTC algorithm but includes an additional optimization step during each round by defining objective functions and constraints as follows:

(6) $$Z_{t1} = \text{Max} \sum_{j,i \in E_{it}} r_{ij}\, x_{ijt},$$ - maximizes the rank of the given course

(7) $$\sum_{i} x_{ijt} \le q_{jt} \quad \forall j,$$ - the number of students assigned to course j in round t does not exceed the number of available seats

(8) $$\sum_{j \in E_{it}} x_{ijt} \le 1 \quad \forall i,$$ - each student is allocated at most one course in a round

(9) $$x_{ijt} \in \{0,1\}, x_{ijt} \in E_{it},$$ - Indicating whether student i is allocated course j in round t (1) or not (0)

(10) $$Z_{t2} = \text{Max} \sum_{j,i \in E_{it}} b_{ij}\, x_{ijt},$$ - maximizes the bids of the given courses

(11) $$\sum_{j,i \in E_{it}} r_{ij}\, x_{ijt} = Z_{t1}.$$ - ensures that the sum of the products of ranks and binary variables equals the maximum value Z_{t1}.

אוניברסיטת
אריאל
בשומרון

SP-O - Second Price Optimization

In the SP-O (Second Price Optimization) algorithm, optimization is performed on course prices during each round by defining the following objective functions and constraints:

$$(12) \quad W_{t1} = \text{Min}\left(Z_{t1}D + \sum_j q_{jt}\, p_{jt} + \sum_i v_{it}\right),$$

$$(13) \quad p_{jt} + v_{it} + r_{ij}\, D \geq b_{ij},$$

$$(14) \quad p_{jt}, v_{it} \geq 0, D \text{ is unrestricted},$$

$$(15) \quad W_{t2} = \text{Min} \sum_j q_{jt}\, p_{jt},$$

$$(16) \quad Z_{t1}D + \sum_j q_{jt}\, p_{jt} + \sum_i v_{it} = W_{t1}.$$

minimize the weighted summation of course prices. It means that we are interested in decreasing the price of courses with fewer seats less than courses with more seats. This objective function, along with (16), which locks (12) at its optimum value, results in finding unique optimum values for the course prices in each round

It is important to note that in order to perform the optimization for the SP algorithm (SP-O), the optimization for the TTC algorithm (TTC-O) must first be completed - Z_{t1} from the computation of TTC-O is needed in order to compute SP-O.

OC - Ordinal-then-Cardinal

The OC (Optimization-Based Comprehensive) algorithm optimizes both TTC and SP algorithms in a comprehensive manner, rather than round by round, by defining the following objective functions and constraints:

(2) $\sum_i x_{ij} \le q_j \quad \forall j,$ - the total number of students allocated to course j does not exceed the available seats

(3) $\sum_j x_{ij} \le k \quad \forall i,$ - each a student can enroll in at most k courses

(4) $x_{ij} + x_{ij'} \le 1 \quad \forall j, j' \text{ with } O_{jj'} = 1,$ - if two courses overlap a, student can get only one of them.

(5) $x_{ij} \in \{0, 1\} \quad \forall i, j.$ - A student is enrolled in course j (1) or not (0)

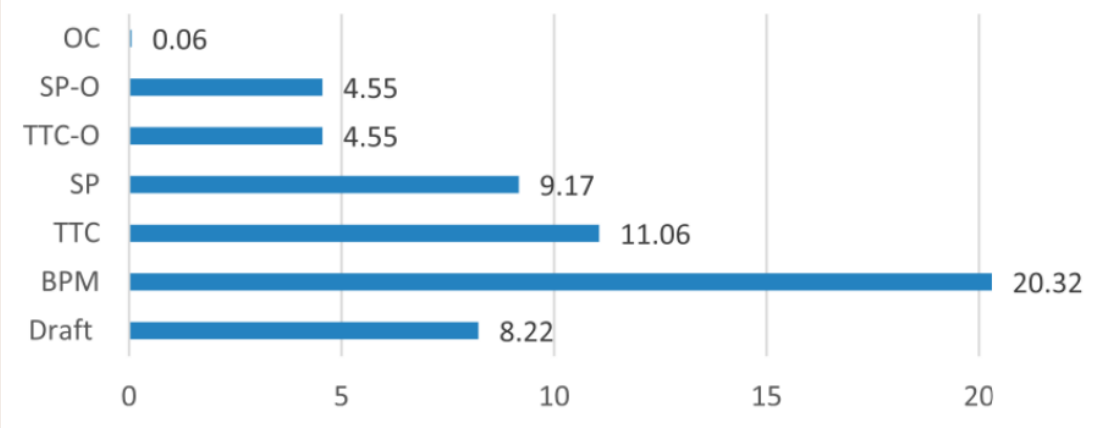
(17) $Z_1 = \text{Max} \sum_{i,j} r_{ij} x_{ij},$ (2)–(5) - maximizes the rank of the total given course

(18) $Z_2 = \text{Max} \sum_{i,j} b_{ij} x_{ij},$ - maximizes the bids of the given courses

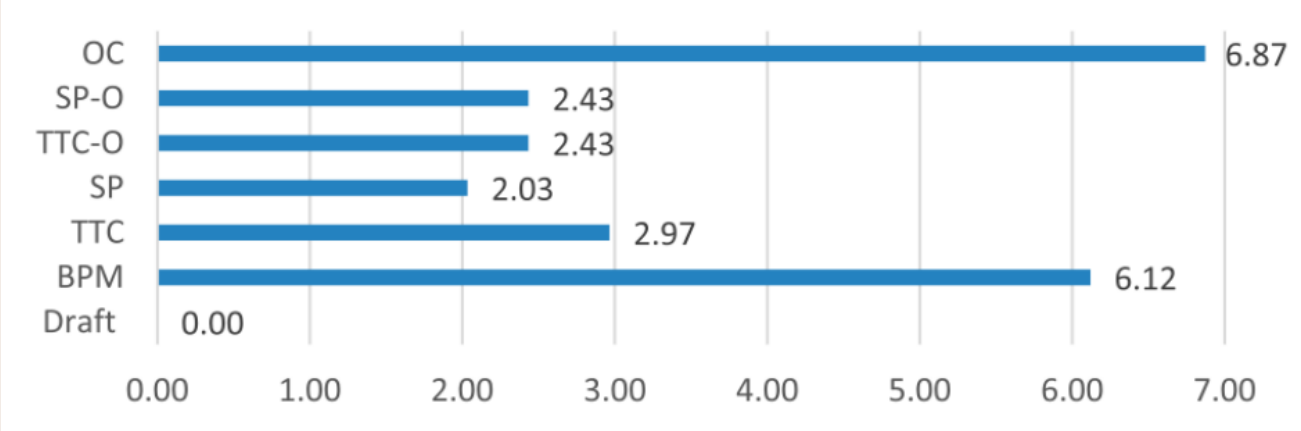
(19) $\sum_{i,j} r_{ij} x_{ij} = Z_1.$ (2)–(5) - ensures that the sum of the products of ranks and binary variables equals the maximum value Z_{t1}.

performance comparison – from the article

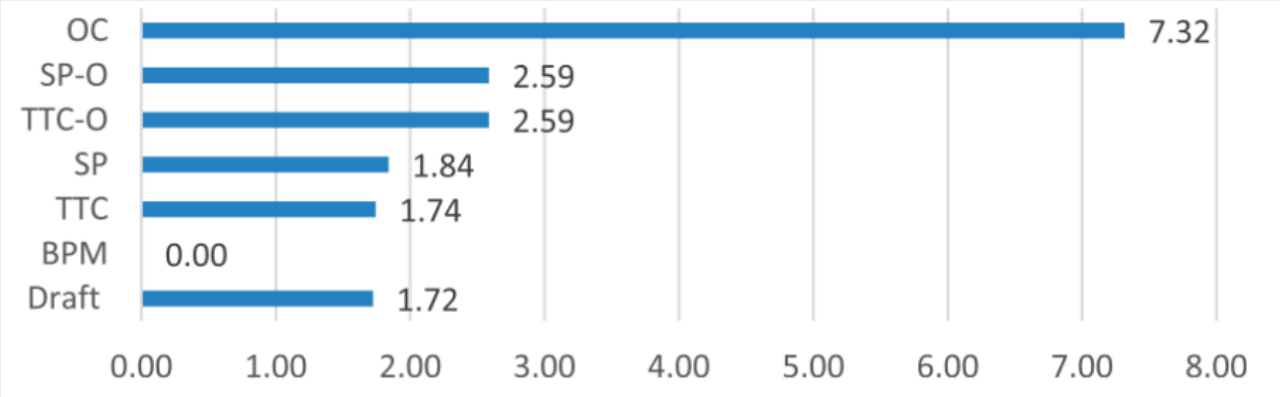
The average number of missed courses:



The average Cardinal utility (total bids): - above Draft



The average Ordinal utility (total ranks): - above BPM



performance comparison – ours

In this section, we compared the performance of our five algorithms (TTC, SP, TTC-O, SP-O, OC) with two existing algorithms from the fairpyx library (iterated_maximum_matching_unadjusted, iterated_maximum_matching_adjusted) across three categories:

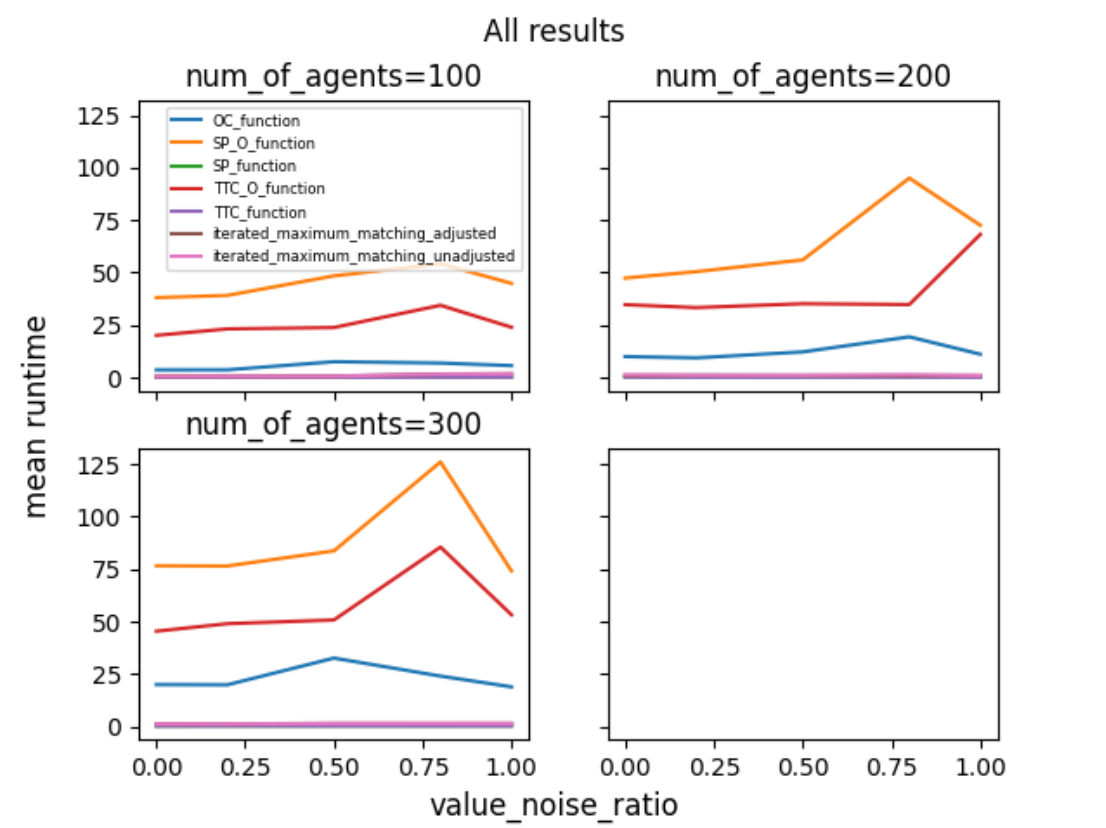
- **Uniform** - Randomly generated data.
- **Ariel** - Data provided by students from Ariel University.
- **SZWS** - Data sourced from a research article.

(https://arxiv.org/abs/2210.00954 , in section 5.5)

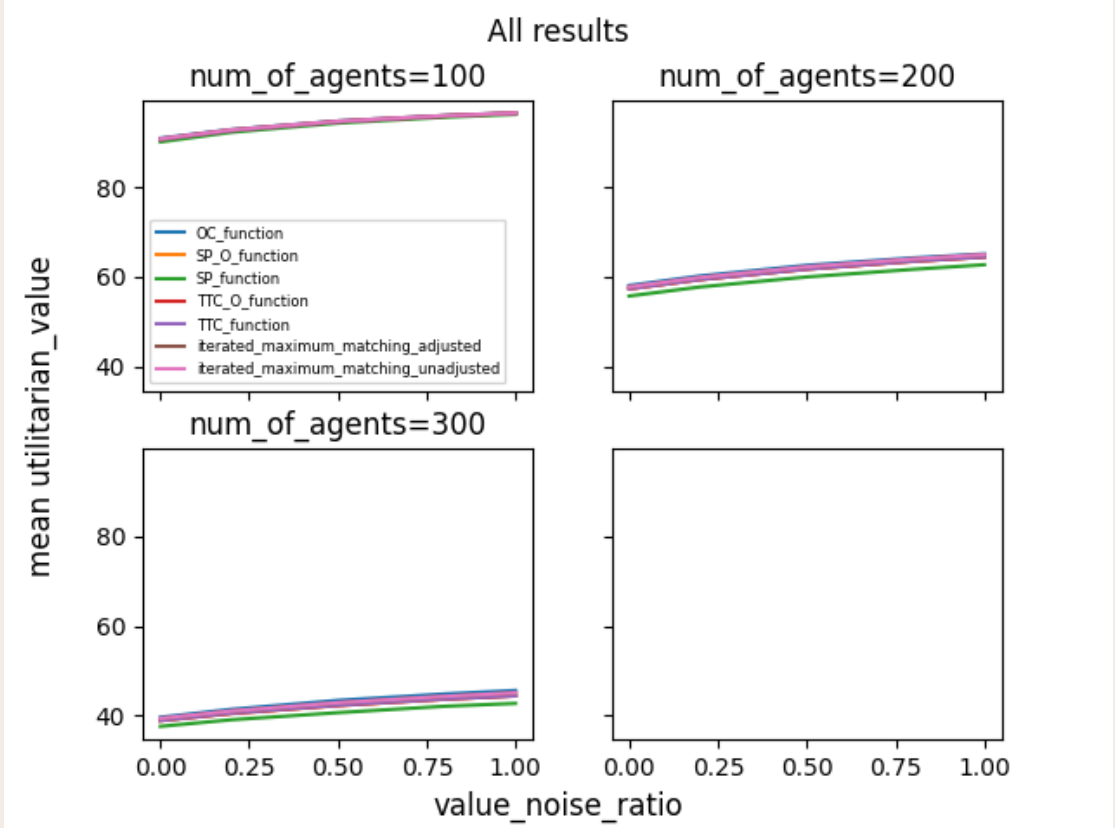
performance comparison – ours

Uniform

runtime:



The average of bids utility:

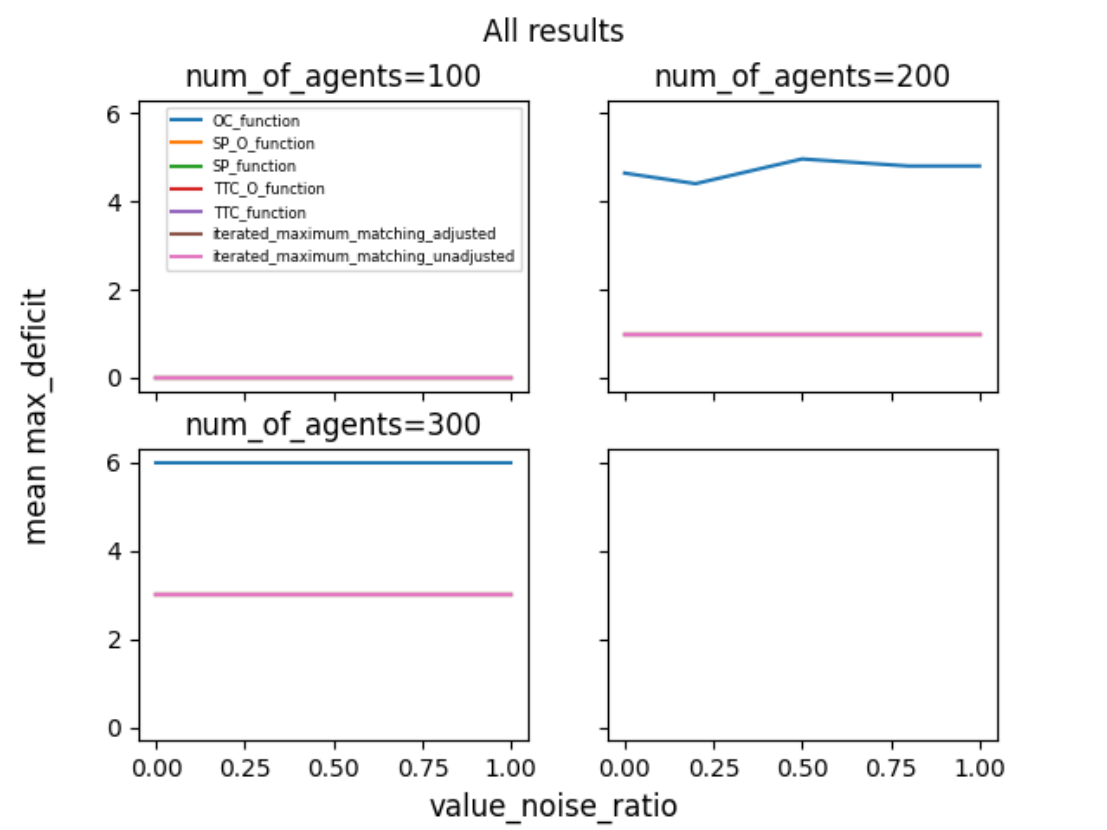


Value noise ratio: the ratio between the value of the courses and the bids students placed on each course—the higher the ratio, the more diverse the course preferences among students.

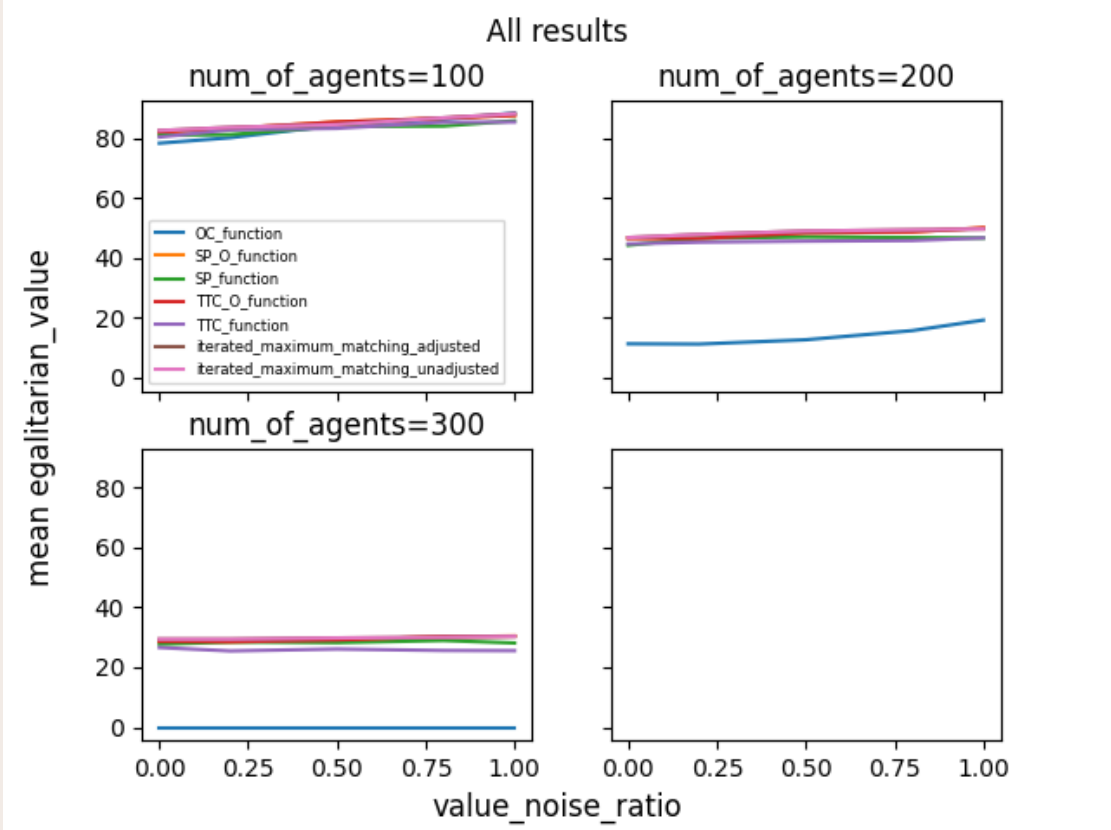
performance comparison – ours

Uniform

The max number of missed courses of a student:



The average of the student with the lowest total bids :

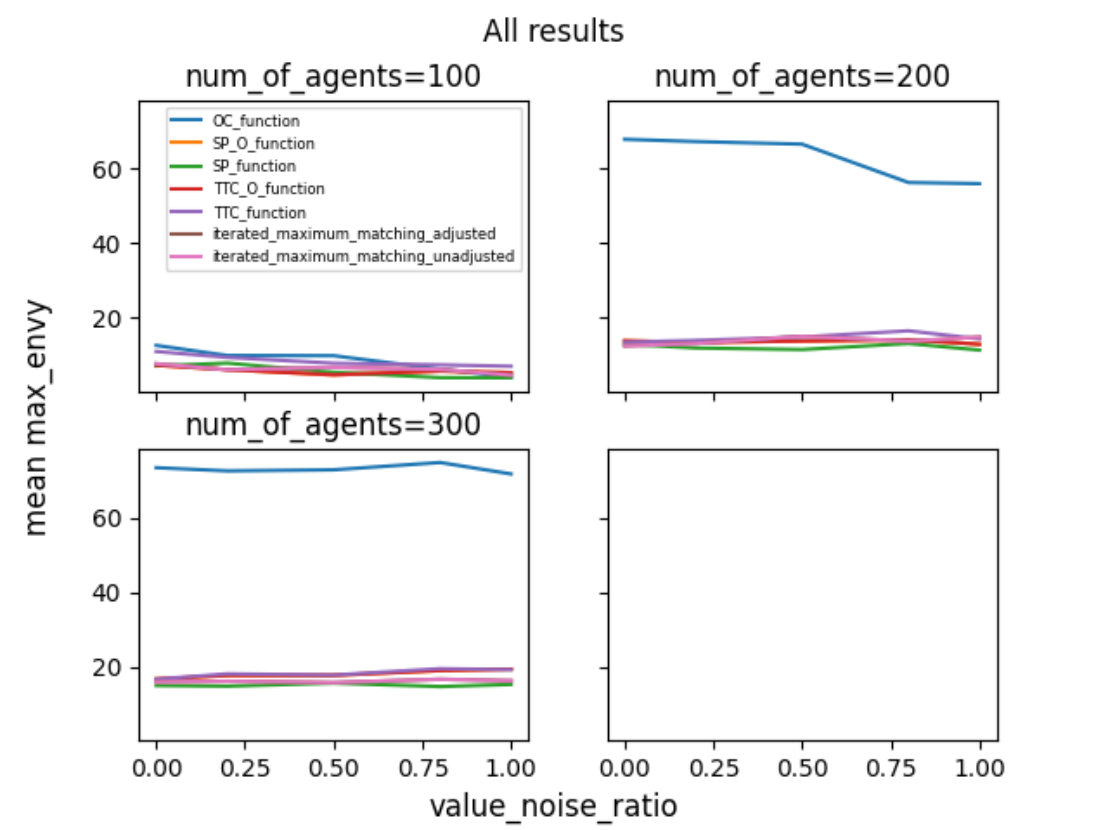


Value noise ratio: the ratio between the value of the courses and the bids students placed on each course—the higher the ratio, the more diverse the course preferences among students.

performance comparison – ours

Uniform

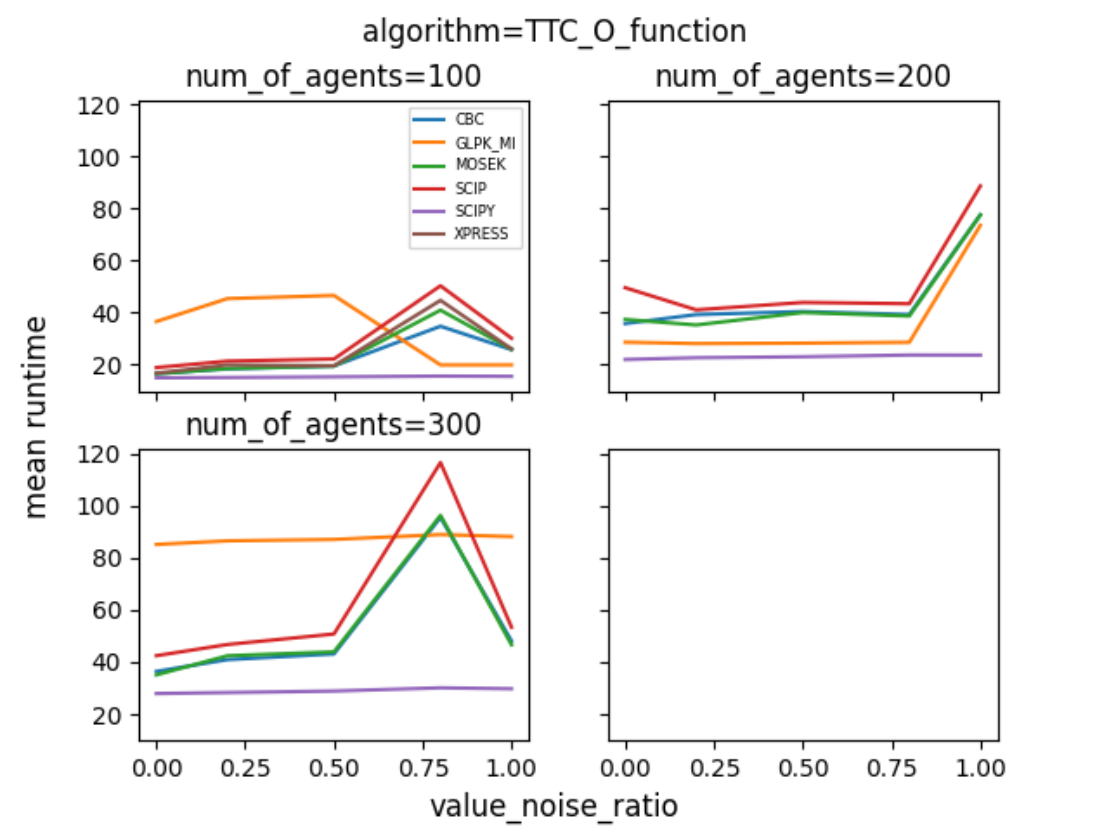
The max number of envy:

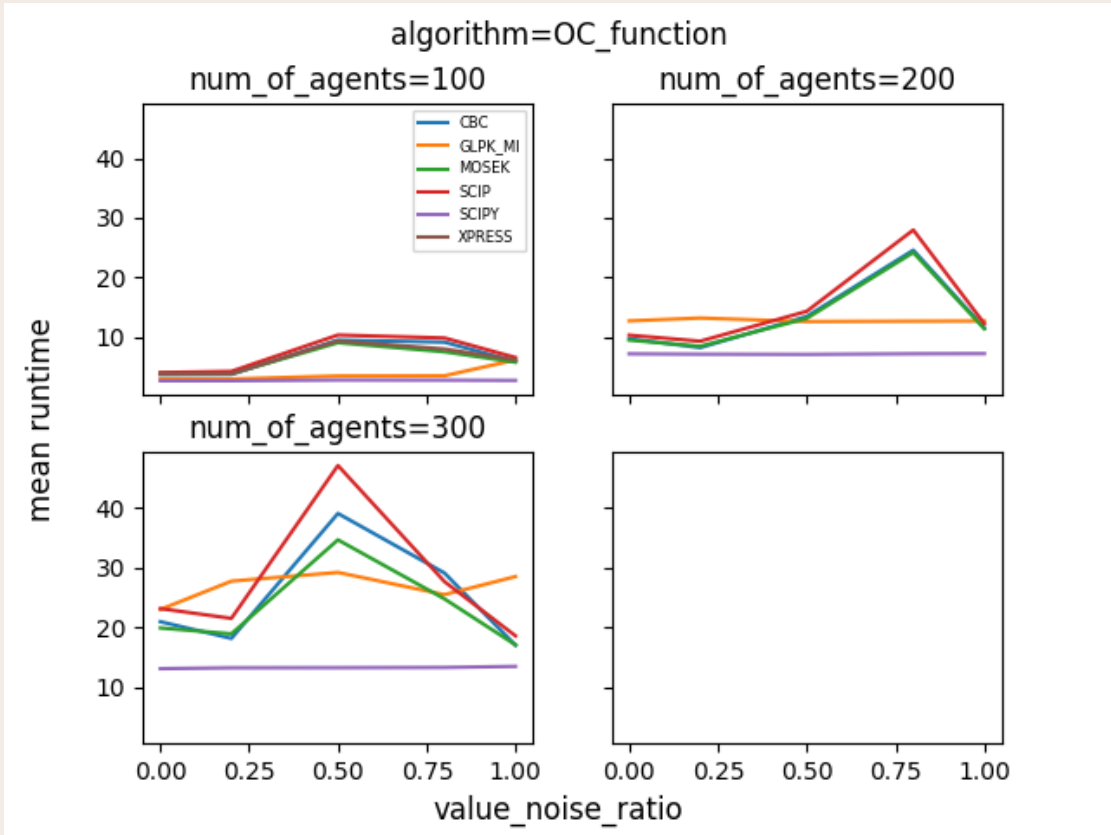


Envy is determined by comparing a student's perceived value of the courses another student received with the value of the courses they actually received. If the perceived value of the other student's allocation exceeds their own, the student feels envy. otherwise, they do not.

Value noise ratio: the ratio between the value of the courses and the bids students placed on each course—the higher the ratio, the more diverse the course preferences among students.

Enhanced Performance





Research conclusions

conclusions

Limitations of the OC Algorithm

OC algorithm expected to be most efficient based on the article.
Indeed, the OC was the most efficient in run time and bids utility maximization,
but over all the TTC-O algorithm outperformed OC.

Envy Analysis:

The article did not address the issue of envy. Our analysis revealed significant drawbacks in the OC algorithm's handling of envy compared to other algorithms.

Deficit Analysis:

The significant gap observed with the OC algorithm prompted a thorough review of our implementation. After an in-depth analysis, we confirmed that our implementation was correct. The findings indicate that, in this context, the OC algorithm performs notably less effectively than the other algorithms.

Conclusion:

Contrary to our initial observation, the choice between TTC-O and OC algorithms depends on the specific requirements of the scenario. It is not possible to definitively state that one algorithm is superior to the other universally.

Research conclusions

difficulties

Challenges with the SP-O Algorithm

Identical Results:

SP-O and TTC-O algorithms produced identical results in both the article and our tests. This raises the question**: What is the significance of the SP-O algorithm?**

Price Calculation Issue:

SP-O's linear programming approach includes course price calculations.
Our in-depth examination revealed that calculated course prices were extremely small- rounding to zero.
This results in prices having no effect on the algorithm's outcomes and merely increases running time.

Conclusion:

The consistent zero price suggests that the results of SP-O and TTC-O are inherently similar, questioning the added value of the SP-O algorithm.

Next Steps:

We have reached out to the authors for clarification but have not yet received a response.

Research conclusions

Lessons learned

- **Challenged Assumptions**: Initial confidence in the OC algorithm's superiority was questioned by our findings, highlighting the need to reassess published conclusions.
- **Communication Challenges**: Issues with author communication underscored the importance of independent problem-solving and collaboration.
- **Fairness Insights**: Investigating envy within algorithms provided new perspectives on fairness and broadened our understanding.
- **Learning Experience**: The project enhanced our problem-solving skills and understanding of algorithmic efficiency and fairness, guiding our future work.

אוניברסיטת
אריאל
בשומרון

THANK YOU FOR LISTENING!

Moriya Ester Ohayon
Tamar Bar-Ilan
Ofek Kats

project advisor: Prof. Erel Segal-Halevi

Based on "Optimization-based Mechanisms for the Course Allocation Problem",
by Hoda Atef Yekta, Robert Day (2020, https://doi.org/10.1287/ijoc.2018.0849)