



MARKET DESIGN PROJECT: C.A.S.T.L.E.

Capacity-Aligned Stable Teaching Location Engine
Tackling classroom allocation in Universities

December 2024

Matthieu MINGUET



EXECUTIVE SUMMARY

The allocation of classrooms in universities represents a complex challenge that currently relies on manual processes and basic criteria like class size, leading to inefficiencies and suboptimal matching between courses and rooms.

This project presents C.A.S.T.L.E. (Capacity-Aligned Stable Teaching Location Engine), a market design mechanism that automates classroom allocation through efficient preference matching. It achieves this through a modified Deferred Acceptance algorithm that matches courses to room types (one to many matching) based on stated preferences while ensuring capacity constraints are met. Key features and results include:

- Theoretical guarantees of stability and Pareto efficiency within room types, and limited manipulability encouraging honest preference reporting
- Computational efficiency with $O(\text{number of courses} \cdot \text{number of rooms})$ average-case complexity (milliseconds for typical medium size university scenarios)
- High performance in simulations: $> 95\%$ allocation rate with over a large proportion of courses matched to top preferences.
- Empirical validation of preferences existence and compatibility with the system through surveys at École Polytechnique.
- Maximization of the utilization of most-requested rooms in terms of use (100%), and capacity ($> 80\%$).
- Minimal faculty input required: 6 ranked preferences sufficient for optimal results, and rare fringe cases requiring manual intervention.

The system represents a significant improvement over current manual allocation methods, offering a scalable, transparent solution that could benefit over 5,000 universities worldwide. C.A.S.T.L.E.'s modular design allows for customization to specific needs while maintaining its core properties. This system is much needed, easy to use and deploy, and has very good market design properties.

CONTENTS

Executive Summary	2
Introduction	4
1 Due Diligence	5
1.1 Current System at Ecole Polytechnique	5
1.2 Current system at Clermont School of Business (Clermont SB)	6
1.3 Market Sizing and existing solutions	7
2 Proposed Solution : C.A.S.T.L.E.'s core principles	8
2.1 Matching algorithm implemented	9
2.2 Properties of the algorithm	9
3 Implementation and Testing	12
3.1 Preference Elicitation for Ecole Polytechnique	12
3.2 Simulated strategies for courses	13
3.3 Results and Analysis	13
4 Effects of the C.A.S.T.L.E. system and next steps	16
4.1 Maximum Utilization of Resources	16
4.2 Quickness and modularity of the system	17
Conclusion	18

INTRODUCTION

The idea for this project came from a simple yet telling experience: a professor struggling at the front of a classroom, trying to juggle between a whiteboard and a projector screen that can't be used simultaneously. In the meantime, I think back to my physics professor in *Classe Préparatoire* who would advocate for a traditional blackboard and chalk over anything else. These contrasting needs highlight a fundamental challenge in academic institution: different courses and teaching styles require different classroom environments.

The inefficiencies in current room allocation systems become apparent when observing these frictions. But beyond teaching tools, the geographical location of the room relative to both students and faculty, proximity to amenities, and layout impact the teaching and learning experience.

These observations point to a deeper market design challenge: how can we efficiently match courses to classrooms while accounting for diverse preferences, physical constraints, and institutional requirements? Current allocation systems, typically managed through manual spreadsheets or basic scheduling software, fail to capture these nuanced preferences and often result in suboptimal matches. The challenge is further complicated by the expanding size and complexity of modern universities, with multiple programs competing for limited space resources.

This project proposes a novel solution to this problem: C.A.S.T.L.E., a matching mechanism that aims to apply market design principles to create more efficient and satisfying matches between courses and classrooms, while ensuring its ease of use and deployment.

We will first **analyze the current systems** in place at Ecole Polytechnique and Clermont School of Business, identifying their flaws and limitations, then doing a market sizing and reviewing the leading academic paper on the subject. Then, we will **present the core principles** of C.A.S.T.L.E., the matching algorithm implemented, and its properties. Finally, we will detail the **implementation and testing** of the algorithm, and the results of a pilot study at Ecole Polytechnique.

1

DUE DILIGENCE

1.1 CURRENT SYSTEM AT ECOLE POLYTECHNIQUE







X currently operates a centralized room allocation system managed by the Registrar's office through a shared spreadsheet. While the basic objective of assigning rooms to all classes is met, the system faces significant operational challenges. The manual allocation process is time-consuming, requires frequent bargaining between departments, and often fails to meet teacher preferences and specific classroom needs. Moreover, it does not provide a reporting of what the faculty values for their classrooms.

The system's current stability relies heavily on the natural distribution of class sizes, particularly in the *Cycle Ingénieur* program where enrollment decreases from 550 students in first year to increasingly smaller specialized groups through second and third year. This pattern allows for predictable allocations, such as Poincare for large lectures and smaller rooms for tutorials, especially when third-year students are off-campus. With the inversion of tutorials and lectures times based on the year, the system can support the needs of the program.

However, mounting pressure from expanding masters and bachelor programs is straining this delicate balance. The competition for rooms, especially between second and third-year needs, is intensifying. While the system functions now due to fortunate enrollment patterns, it lacks the robustness to handle future growth and increasing complexity.

1.1.1 • FLAWS IN THE CURRENT SYSTEM

In terms of market design, this system is a centralized dictatorship, as in a control economy where resources are allocated based on the sole criteria of class size. It's flaws which we will detail in the following table:

Flaw	Description	Intensity
Information Problem	<ul style="list-style-type: none"> No preference elicitation Manual spreadsheet means no way to communicate preferences, constraints, or trade 	▼ 
Matching Mechanism	<ul style="list-style-type: none"> Single-Criterion Optimization: capacity matching Manual Coordination Cost: bargaining No formal Trading Mechanism: no beneficial trades 	▼ 
Incentive Problems	<ul style="list-style-type: none"> No strategic reporting: no formal way to express the preferences for rooms or features Lack of Price Mechanism: no way of quantifying preferences 	▼ 
Scalability	<ul style="list-style-type: none"> Brittle equilibrium: program change or growth could pose a threat to the current system Manual processing bottleneck 	▼ 
Efficiency problem	<ul style="list-style-type: none"> No Optimization Algorithm Suboptimal Outcomes: room for Pareto improvements Resource Utilization: inefficient use of features and capabilities 	▼ 
Adaptation and flexibility	<ul style="list-style-type: none"> Rigid system: changes mid-term can be difficult Limited feedback loop: for allocation and features 	▼ 

As we can see, while the current system has major flaws like the lack of a matching mechanism, it still



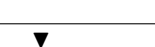
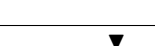


ensures all teachers have a room, and the school has functioned with this system for the past few years without any issues, especially when the *Cycle Ingénieur* program was the only program at the school.

1.2 CURRENT SYSTEM AT CLERMONT SCHOOL OF BUSINESS (CLERMONT SB)

The Clermont SB has a similar system to Ecole Polytechnique, with a centralized room allocation and planning system. A dedicated member of staff is responsible for managing the system in place: Program Directors submit room requirement and planning forms to the Planning Office, which then manually attributes the rooms based on the information provided, and the availabilities of the rooms. These types of requirements include the number of rooms, the equipment, expected cohort size, and if the rooms need to be close to each other or not (for example for group work). The number of variables makes it very difficult to find a solution satisfying all requirements of all courses. While the system allows for inputs from the faculty, the major issue is that the large number of programs and the increasing number of students in each program is making the system increasingly complex to manage by hand and is leading to inefficiencies and tensions in the allocation. Moreover, this system is quite slow and requires a large amount of time to manage, which could be streamlined and better used elsewhere. Finally, the system is not transparent, and the faculty does not have a clear understanding of how the rooms are allocated, while it also doesn't encourage truthful reporting of preferences.

1.2.1 • FLAWS IN THE CURRENT SYSTEM

Similarly as what we did for Ecole Polytechnique, we have identified the following flaws in the current system at Clermont SB:

Flaw	Description	Intensity
Information Problem	<ul style="list-style-type: none"> Preference reporting through requirement forms Lack of transparency in allocation process 	
Matching Mechanism	<ul style="list-style-type: none"> Manual attribution based on submitted forms Single-person decision making process No formal mechanism for resolving conflicts 	
Incentive Problems	<ul style="list-style-type: none"> No incentive for truthful preference reporting Lack of feedback mechanism 	
Scalability	<ul style="list-style-type: none"> Growing complexity with increasing programs Rising student numbers strain current system Single point of failure (dedicated staff member) 	
Efficiency problem	<ul style="list-style-type: none"> Time-intensive manual process Inefficiencies in allocation Resource under-utilization due to information gaps 	
Adaptation and flexibility	<ul style="list-style-type: none"> System struggles with program diversity Difficult to make mid-term adjustments Slow response to changing needs 	

As we can see, while this system offers more inputs from the different stakeholders, it is limited by the fact that it has to be manually run, and an increasing number of programs and students compared to the number of rooms, or even more rooms could cause issues regarding the length of the process and the fairness of the allocation.

1.3 MARKET SIZING AND EXISTING SOLUTIONS

1.3.1 • MARKET SIZING

There are around 18500 universities worldwide, representing around 250m students. However, a new system such as C.A.S.T.L.E. would be more suited to larger universities, especially in developed countries, where the number of students is higher, and programs are more diverse. As such, we estimate a theoretical market size of 5000 universities, mainly in Europe and North America. At first, this system would target around 80 universities in these markets, which considering their size could already impact upwards of 200000 students. Thus, C.A.S.T.L.E. could bring an improvement over the current systems in place in these universities, and potentially be a game-changer in the way universities allocate their resources.

1.3.2 • EXISTING PAPERS

Classroom allocation or time-tabling is a well-studied problem in the literature, with many papers proposing different solutions to the problem. For example, Rivero, Escárcega, and Velasco (2021) proposed an Integer Linear Programming model for classroom assignment to tackle this issue, while other solutions focus on graph coloring algorithms or genetic algorithms to allocate classrooms and timetables. In almost all cases, the solution to solve both timetabling and classroom allocation is to decouple the two problems and solve them sequentially, as the classroom allocation algorithm generally output near 100% utilization of the rooms, which we will see is also the case for C.A.S.T.L.E.. The basic formulation of the problem for classroom allocation is as follows:

- Let x_{ij} be a binary variable where i the classroom, j the course and : $x_{ij} = 1$ if course j is assigned to classroom i , 0 otherwise.
- The objective function is : $\min \sum_{i,j} c_{ij}x_{ij}$ where c_{ij} is the cost of assigning course j to classroom i .
- c_{ij} represents the cost of assigning class j to classroom i , and can incorporate functions like room size mismatch, distance from the professor's department, equipment mismatch, etc.

The constraints are generally the following:

- Single assignment constraint: $\sum_i x_{ij} = 1$ for all j
- Room capacity constraint: $\text{enrollment}_j \leq \text{capacity}_i \forall x_{ij} = 1$

In cases where timetables are also considered, the time t is added as a third dimension to the problem, and the constraints are modified to include the time dimension (professor, room and student availabilities) This formulation guarantees an optimal solution (if solvable) and is able to balance different objectives by weighting the c_{ij} , while being very flexible in terms of constraints and objectives.

However it is an NP-hard problem to solve and the solution space grows exponentially with the number of rooms, classes (and time slots). As an example, the Ecole Polytechnique has 76 rooms, around 65 to 70 classes at one time, for a complexity of $76^{70} \approx 10^{140}$, which is infeasible to solve with a brute force algorithm, and we have not even considered the time dimension. Moreover, the model is static and can't easily handle dynamic changes.

Work exists to improve upon this by for example relaxing the strict constraints and adding a violation penalty, doing pre-processing to reduce the search space, or using heuristics to find a good solution in a reasonable time. Other systems such as graph coloring exist, but they are also NP-hard, like the Integer Linear Programming model, and are generally less flexible as they have difficulty incorporating different objectives and soft constraints.

In conclusion, while significant academic work has been done on the subject of classroom allocation and timetabling, the problem remains complex and difficult to solve, especially in a dynamic environment like a university. Findings from these papers are useful to understand the problem and the constraints, but a new

approach is needed to tackle the problem in a more efficient way. We do keep in mind the idea of decoupling the timetabling from the allocation, as C.A.S.T.L.E. only focuses on the allocation part of the problem.

2

PROPOSED SOLUTION : C.A.S.T.L.E.'S CORE PRINCIPLES

The key objectives of C.A.S.T.L.E. in light of the current systems and the existing literature are the following:

- **Efficiency:** Maximizes room utilization and stakeholder satisfaction through optimal allocation of classrooms to courses.
- **Computability:** Delivers solutions within reasonable time frames on standard hardware, even at university scale.
- **Flexibility:** Adapts to diverse objectives and constraints while accommodating institution-specific requirements.
- **Transparency:** Ensures allocation decisions are understandable and promotes honest preference reporting.
- **Reliability:** Minimizes manual interventions by consistently producing near-complete room assignments across complex scenarios.

Moreover, we should also try to limit the time spent by faculty on this process, as it takes away from their time. C.A.S.T.L.E. should also not be based on any currency as this could cause problems with faculty and require more thought and time to use the system. Based on the objectives of fitting the cohorts in the rooms and allowing the maximum amount of stakeholders such as faculty, students, and outside speakers to benefit from the best and most sought after rooms, a cardinal parameter is to consider the "fill amount" of the rooms, or in other terms, to minimize $\text{room}_{\text{capacity}} - \text{course}_{\text{cohortsize}}$.

To achieve these objectives, two main algorithms were initially thought of:

YOU REQUEST MY ROOM, TAKE IT IF YOU'RE BIGGER (YRMRTIYB)

This is based on a similar idea as *You request my house I get your turn* which is already based on the Top Trading Cycles (TTC) algorithm. Each course would point towards the room they want, and the rooms towards their top choice of course. Then, the algorithm would find cycles of courses and rooms, and allocate the rooms to the courses in the cycle. courses then point towards their next choices, and the algorithm continues until someone requests a room that is already taken, at which point if their cohort size is bigger, they get the turn of the first course. Then the algorithm runs once more, and the cycle continues until all rooms are allocated.

The main problem of this algorithm is that stability is not guaranteed, which is a prerequisite for a long term allocation process. It does however bring strategy-proofness which is desirable to avoid strategic reporting.

DEFERRED ACCEPTANCE ALGORITHM (DA)

In this case, faculty members (or inversely rooms) would propose to a room (or inversely a faculty member) based on their preferences following the classic deferred acceptance algorithm. This approach would guarantee stability, albeit without being inherently strategy-proof. Additionally, it is a well-know algorithm that has been implemented in many different contexts. The final version of C.A.S.T.L.E.'s algorithm is similar to that of "the Match" for medical students as it is a one-to-many matching.

2.1 MATCHING ALGORITHM IMPLEMENTED

Room of identical types, such as rooms in the same corridors having the same equipment are grouped into a single type that represents them, order to minimize the number of choices having to be made by faculty members. As such their **preferences are for room types**. For the core algorithm we consider the fit function f , chosen by the administration, in its most basic form : $\text{room}_{\text{capacity}} - \text{course}_{\text{cohortsize}}$. Before running the algorithm, we need to know the following parameters: $\text{room}_{\text{type}}$, $\text{room}_{\text{capacity}}$, $\text{course}_{\text{cohortsize}}$, and $\text{course}_{\text{preferences}}$.

1. Courses propose to their top choice of room type based on their preferences. We then run the steps 2 to 4 for each room type separately.
2. All proposals to a type are grouped, including any courses already tentatively allocated in the rooms.
3. In order of increasing capacity, rooms tentatively choose the course corresponding to $\min_{\text{course}} f(\text{course}, \text{room})$, and it is removed from the list of proposals.
4. Once all rooms in the type have chosen the remaining courses are sent back to the global proposal pool, or there are no remaining proposals in this type.
5. The remaining courses change their current choice of room type to the next one and go back to step 1.
6. The algorithm stops when all courses have been allocated a room, or the remaining courses have tried all their choices.

2.2 PROPERTIES OF THE ALGORITHM

The proposed algorithm has multiple desirable properties for our market design problem.

2.2.1 • DEFINITIONS AND NOTATION

Let us define:

- $C = \{c_1, \dots, c_n\}$: Set of courses
- $R = \{r_1, \dots, r_m\}$: Set of rooms
- $T = \{t_1, \dots, t_k\}$: Set of room types
- P : the maximum number of preferences per course ($P \leq T$)
- \succ_c : Preference relation for course c over room types
- μ : A matching function $C \rightarrow R \cup \{\emptyset\}$
- $s(c)$: Size of course c
- $\text{cap}(r)$: Capacity of room r
- $\text{type}(r)$: Type of room r

2.2.2 • COMPUTABILITY

The allocation process can be visualized as the following flowchart:

1. For each unmatched course ($O(C)$):
 - Group proposals by type: $O(C)$
 - For each room type ($O(T)$):
 - Sort rooms by capacity: $O(R \log R)$
 - For each room, evaluate all available courses: $O(R \cdot C)$

The algorithm has a worst-case time complexity of: $O(C \cdot T \cdot (R \log R + R \cdot C))$ and a space complexity of $O(R + C)$.

However, in practical applications, several factors contribute to better average-case performance:

1. Most courses are matched in early iterations, reducing the effective number of rounds from $O(C)$ to approximately $O(P)$
2. Room types T is a small constant (approximately 10 in our implementation), and $P \leq T$.
3. With a relatively small number of rooms ($R < 1000$), the room sorting component becomes nearly linear: $O(R \log R) \approx O(R)$

As such, the result is an average-case time complexity of simply $O(C \cdot R)$, and space complexity $O(R + C)$. This is a significant improvement and enables the algorithm to run in a reasonable time frame on standard hardware, even at university scale. Empirical evidence from our implementation supports this analysis. With a typical scenario of 70 courses and 70 rooms and spread out preferences, the algorithm completes in 150-300 allocation steps, with most courses being matched within their first three preferences.

2.2.3 • STABILITY ANALYSIS

Theorem 1 (Stability). *C.A.S.T.L.E. allocation mechanism produces a stable matching.*

Proof. Suppose, for contradiction, that the matching μ is unstable. Then there exists a blocking pair (c, r) where: c prefers r to its current assignment $\mu(c)$ and r prefers c to its current occupant $\mu^{-1}(r)$. However, by construction of the algorithm:

1. If $\text{type}(r) \succ_c \text{type}(\mu(c))$, then c would have proposed to rooms of type $\text{type}(r)$ before $\text{type}(\mu(c))$
2. Within each type, rooms are assigned to best-fitting courses based on capacity utilization
3. If $s(c) \leq \text{cap}(r)$ and $\text{capafit}(r, c) < \text{capafit}(r, \mu^{-1}(r))$, then r would have been assigned to c in the per-type processing phase

Thus contradicting the existence of a blocking pair, proving stability. □

2.2.4 • PARETO EFFICIENCY

Theorem 2 (Pareto Efficiency). *C.A.S.T.L.E. allocation produces a Pareto efficient matching among stable matchings within each type.*

Proof. Consider a matching μ produced by the algorithm. Suppose, for contradiction, that there exists another stable matching μ' that Pareto dominates μ . This means:

1. $\forall c \in C: \text{type}(\mu'(c)) \succeq_c \text{type}(\mu(c))$

2. $\exists c^* \in C: \text{type}(\mu'(c^*)) \succ_{c^*} \text{type}(\mu(c^*))$

For types processed in round k , let C_k be the set of courses assigned in that round. The algorithm maximizes:

$$\sum_{c \in C_k} \text{capafit}(c, \mu(c))$$

Therefore, any improvement for c^* would necessarily worsen the fit for some other course in the same round, contradicting Pareto dominance. \square

2.2.5 • STRATEGY-PROOFNESS ANALYSIS

While the classic deferred acceptance algorithm is not generally strategy-proof, C.A.S.T.L.E.'s variant has interesting properties that make it difficult to manipulate.

Theorem 3 (Limited Manipulability). *Capacity constraints, Multi-dimensional preferences, Information asymmetry limit manipulability of the mechanism.*

Proof. Consider a course c attempting to manipulate its preference list \succ_c . For successful manipulation:

1. c must know:
 - Preferences of other courses: $\{\succ_{c'}\}_{c' \in C \setminus \{c\}}$
 - Sizes of other courses: $\{s(c')\}_{c' \in C \setminus \{c\}}$
 - Room capacities: $\{\text{cap}(r)\}_{r \in R}$ which he does know as it is public information
2. c must find a preference list \succ'_c such that:

$$\text{type}(\mu'(c)) \succ_c \text{type}(\mu(c))$$

where μ' is the matching under manipulated preferences

\square

Practical Implications In conclusion, theoretical properties have important practical implications:

1. Stability ensures no course-room pair has incentive to deviate and should allow for a long-term deployment
2. Pareto efficiency guarantees no wasteful assignments within types, maximizing room utilization and stakeholder satisfaction
3. Limited manipulability reduces gaming concerns in practical deployments, promoting honest preference reporting and makes it easier to understand the preferences of the faculty
4. Thus, the mechanism balances competing objectives:
 - Course preferences over room types
 - Efficient room capacity utilization
 - Fair allocation among courses

Hence, the proposed mechanism is well-suited to the challenges posed by university classroom allocation, offering a robust, efficient, easy to use solution, that offers good properties for the stakeholders. It manages to achieve computability by restricting the tested combinations to those asked by the faculty in the Gale-Shapley algorithm, which also guarantees stability and Pareto efficiency. Finally, it is not strategy-proof, but the limited manipulability encourages honest reporting of preferences and reduces the risk of gaming the system.

3

IMPLEMENTATION AND TESTING

The algorithm was implemented in Java, leveraging object-oriented programming principles to model the problem and solution effectively. Moreover, it allows for clear organization of the code and easy extension and modification of the algorithm with different parameters, fit functions, rooms or constraints through the use of abstract classes and interfaces. This creates a modular and flexible environment for coding and testing the algorithm. Most importantly, the execution speed of Java versus Python is nearly 100 times faster, which eases the use of the algorithm on a large scale and quick changes.

This is achieved without the need for complex parallelization or optimization, as the algorithm is already efficient enough to run on a standard laptop. On a scenario of the size of Ecole Polytechnique, the algorithm runs in around 10 milliseconds, which is more than acceptable for a near real-time allocation process.

The code structure and key elements are detailed in the annex. The main classes are the allocation algorithm, all the objects representing the courses, rooms, and room types, and the fit function.

3.1 PREFERENCE ELICITATION FOR ECOLE POLYTECHNIQUE

In order to establish the feasibility and the estimated impact of the C.A.S.T.L.E. system, a pilot study was conducted at Ecole Polytechnique on the preferences of stakeholders. Thus, a survey was sent out to students to gather information on their preferences for rooms. For students, the survey was a ranking of the different room types based on their preferences.

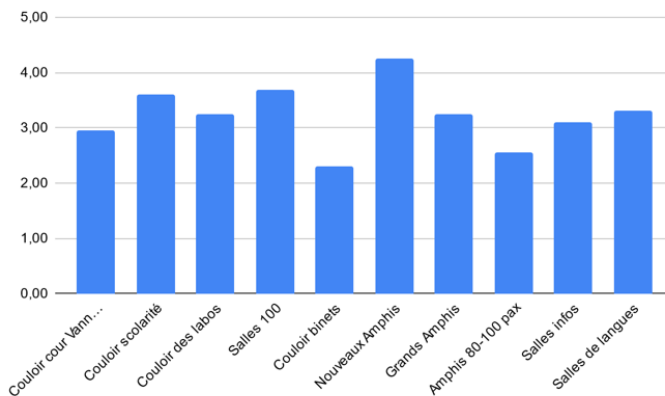


Figure 1: Survey results for Ecole Polytechnique

The rooms were divided into 10 types, based on their location, equipment, and capacity. The survey was sent out to 70 students, 33 of which responded. At Ecole Polytechnique, rooms in the same corridors (or wing) have the same equipment and roughly the same size, leading to an easy grouping of the rooms into types.

The results show that students have clear preferences for certain room types, as grading of the rooms varies between 2.2 to 4.1 out of five. With the results grouped around 3, the standard deviation is quite low. This implies that preferences exist but as average – standard deviation < 1 , they are not too strong. Therefore, not having your top choice while not ideal would not be a game-breaking issue. In conclusion, this means that a classroom allocation optimization is warranted

These preferences are driven by four major differentiation factors that play a role in the learning experience and the quality of life of students and faculty. Room equipment represent the first key driver, from screens, to boards, a good projector, the provision of enough power outlets for students, or even the tables used for the room. The second driver, room size, is critical to accommodate the cohort, and what type of work is done in the room, which brings us to the third driver the room layout, be it inclined, flat, or very long and narrow.

Finally, the proximity to essentials such as the coffee machine, the labs of the professor, or the canteen is also a key driver, encompassed with things such as the number of stairs in our fourth driver: location and accessibility. Overall, there are a variety of factors that affect learning and quality of life, and the preferences of both students and teachers reflect that.

3.2 SIMULATED STRATEGIES FOR COURSES

In order to test the algorithm and deliver some simulated results, the registrar's office at Ecole Polytechnique provided us with the list of the rooms and their capacities. We did not however have the list of the courses and their cohort size, so we had to simulate this data. We used a random distribution of the cohort size, but with parameters that simulate the difference we can have between tutorials and lectures, and the fact that some courses are more popular than others. Therefore, 85% of the courses have a cohort size between 10 and 35, representing tutorials (and the classes of the third years and masters which are highly specialized), while the remaining 15% have a cohort size between 35 and 200, representing lectures.

Additionally, several strategies regarding the preferences of the courses were devised and tested in order to see the impact of the preferences on the final allocation, and test different scenarios:

- **Random:** The courses have random preferences for the room types based on a uniform distribution
- **Smart Random:** The courses have uniform random preferences for the room types, but only if a room with sufficient capacity exists in the type.
- **Size-based:** The courses have preferences based on the average room size of the type, and rank them to minimize the difference between it and the cohort size.
- **Fixed preferences:** The courses have fixed preferences for the room types, based on the survey results (average grade). In that case all courses have the same preferences, in decreasing order of the average grade.
- **Satisfaction:** The courses have preferences based on the pooling of students, so as to represent a real life scenario (with extrapolation to increase the number of courses to an arbitrary number). We therefore have more variation in the preferences expressed compared to the fixed preferences.

Once the strategies were defined, the algorithm was able to run on simulated data and produce results which we will detail in the next section. The other parameter that can vary is the number of preferences expressed, from 3 to 10 (max). Unless stated otherwise, the results are based on the 76 rooms of Ecole Polytechnique, and between 65 and 70 courses, with 20 run averages for each strategy.

3.3 RESULTS AND ANALYSIS

• ALLOCATION RATE BY STRATEGY

The graph below shows the allocation rate for each strategy, with the number of preferences expressed (3, 5 or 10). The allocation rate is defined as the percentage of courses that have been allocated a room at the end of the process.

The Fixed (fixed_10) and Smart Random (smart_random_10) strategies have the highest allocation rates, at 99.7%, where courses are all allocated a room, except for rare cases where there are too many big courses simultaneously. This is to be expected as courses run through all the room types, and therefore have a near 100% chance of being allocated a room. The fixed strategy is our worst-case scenario, as it represents a scenario

where all courses have the same preferences, and the algorithm has to allocate them based on the capacity of the rooms, and a lot of shuffling is needed.

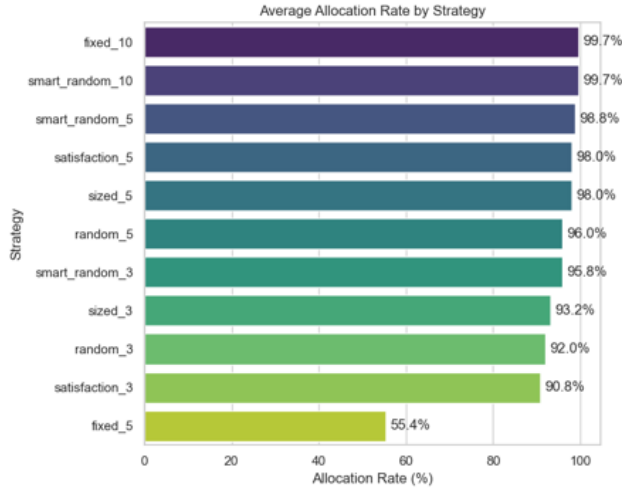


Figure 2: Average Allocation rate over 20 runs by strategy and number of preferences

This is particularly important as it represents the most realistic scenario among all tested strategies. It also indicates that ranking all 10 room types would not be necessary, as 98% of the courses are already allocated using 5 preferences.

One conclusion that can be drawn from this graph is the importance of spreading out preferences for the rooms, as varied choices allow for better allocation rates. However this cannot be done without a variety of room types offering the basic requested features from faculty and students.

There is a clear and expected pattern, that allowing more preferences generally improves allocation rate across all strategies. Random for example improves from 92% to 96% with 5 preferences, which we can explore next.

• IMPACT OF THE NUMBER OF PREFERENCES

The simulations are based on the Satisfaction strategy, as it represents the most realistic scenario. The impact of the number of preferences expressed by the courses on the allocation rate is shown below. The parameters are 70 courses, 76 rooms, and a cohort size between 10 and 200, averaged over 100 runs. For each run, the seed for the random number generator is fixed to ensure coherence of the results.

The theoretical best-case scenario, Size-based (sized_3 and sized_5) where the courses have preferences based on expected room capacity, meaning rooms with capacities closest to the cohort size, performs well even at low preference numbers. The Smart Random strategy, where the courses have random preferences but only if a room with sufficient capacity exists in the type, also performs well at 98.8%, explained by the rooms having spread-out preferences, with no "dead-end" room types that would be too small for them.

The real life scenario (satisfaction_5), performs well at 98%, indicating that when using real life preferences based on student polling, the algorithm can still achieve high allocation rates.

The data shows that allowing more preferences consistently leads to better allocation rates, with a particularly significant improvement as they increase from 1 to 6. When courses can express only one preference (satisfaction_1), the allocation rate is relatively low at 52.7%. However, this rate improves dramatically to 78.8% with two preferences and continues to rise steadily with each additional preference option.

A notable threshold appears at 6 preferences, where the allocation rate reaches 99.1%. Beyond this point, the improvements become marginal, with 7, 8, and 9 preferences all achieving nearly identical rates of 99.2%. This suggests that 6 preferences may represent an optimal balance point between choosing complexity for faculty and allocation efficiency. This indicates a good sign of being able to allocate nearly all courses with a limited number of preferences.

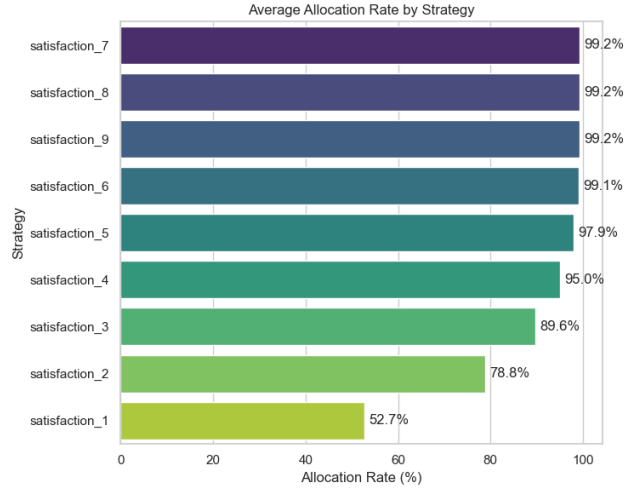


Figure 3: Impact of the number of preferences on the allocation rate

• PERFORMANCE AND SATISFACTION METRICS

The efficiency of the matching process implemented in C.A.S.T.L.E. has been measured by two key metrics: Average choice rank and high-rank rate defined as follows: Average choice rank is simply the average rank of the room type allocated to the courses in its preferences, while the high-rank rate is the percentage of courses that have been allocated a room out of the top three. These metrics are important for the satisfaction of the system, as they show how well the preferences of the courses are taken into account. In both cases, the lower the value, the better the performance.



Figure 4: Average choice rank and high-rank rate by strategy and number of preferences

Here we clearly see the worst cases scenario for the Fixed strategy, where the average choice rank of 4.9 and a high rank rate of 65.9%. This is to be expected as the courses have the same preferences. The focus here

is on the other strategies, none of which go over 2 for the average choice rank, and high-rank rate under 7%. This globally indicates that the algorithm is able to allocate the courses to their top choices, while respecting the capacity constraints and the needs of the administration. The high-rank rate is particularly important, as it shows that there would be very few professors and teaching assistants that would not be satisfied with their room allocation.

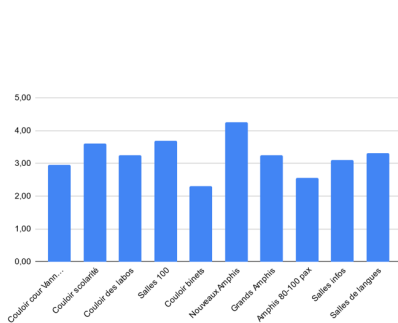
The real life based scenario: satisfaction, has one of the worst performance metrics, with an average choice rank of 1.73 and a high-rank rate of 6.5%. While not the best, this is still a very good result, and shows what C.A.S.T.L.E. can achieve.

4

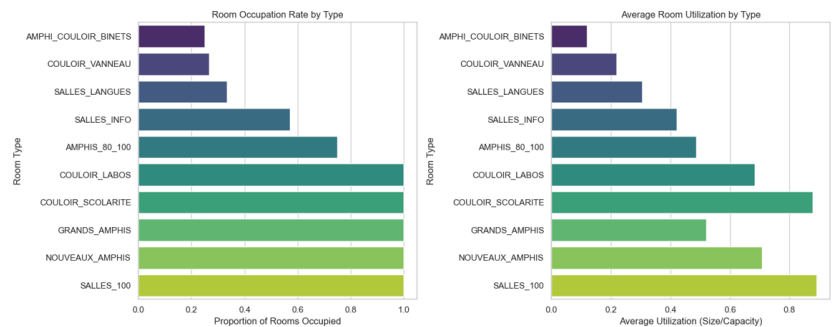
EFFECTS OF THE C.A.S.T.L.E. SYSTEM AND NEXT STEPS

4.1 MAXIMUM UTILIZATION OF RESOURCES

One of the visible effects of the C.A.S.T.L.E. system is the maximum utilization of the most requested resources, as the algorithm is designed to minimize the difference between the cohort size and the room capacity. This is particularly important to allow the best rooms to be used by the most students and thus get the most out of the school investment in these capacities. For the case of Ecole Polytechnique, using the real life scenario, the top 5 room types are used at 100%, and they have very high occupancy rate, as seen in the graph below. The lower occupancy rate for the *Grands Amphis* and *Nouveaux Amphis* is due respectively to being the biggest rooms up to 800 students (while the simulation only goes up to 200 for lectures), and tutorials being simulated to 35 students max while the room can hold 50. Accounting for those two abnormalities, the most requested rooms are used at over 80% of their capacity, which is a very good result and shows the efficiency of the algorithm in allowing more students to benefit from the best rooms.



(a) Recap of the survey results



(b) Percentage of rooms used and occupancy rate

Figure 5: Effect of signaling maximum utilization of resources

For reference, the results of the survey are shown again in the first graph. These results show the strong correlation between the preferences from the poll and the allocation of the rooms which is what we aim for. Less requested rooms are also used, but at lower rates, and the average occupancy is weighed down significantly by the empty rooms.

Overall, the C.A.S.T.L.E. system allows for a better allocation of the rooms, and a better use of the resources, which is particularly important in a university setting where the number of rooms is limited.

4.2 QUICKNESS AND MODULARITY OF THE SYSTEM

As explored in the implementation section, the algorithm is very quick to run, and can be easily modified to take into account different parameters, by modifying the fit function of the rooms. The high allocation rate ($> 95\%$) and low average choice rank (< 2) implies that the algorithm needs little to no manual intervention to account for the fringe cases where courses would end up not allocated to a room. Moreover, the rooms in the types being ran in order of increasing capacity, which might be counter-intuitive, makes it so that any room left empty will be bigger and thus an extra course would be easier to fit in. Most importantly, the Deferred Acceptance algorithm is stable, so no courses would ask for a room exchange, and modifying slightly the entries of the algorithm should not change the results significantly.

If there are any fringe cases, two solutions, either the Registrar's office manually allocates that room, or if possible extend the preferences of the courses to the maximum number of room types, which allows 100% allocation rate if there aren't more courses than rooms. Overall, C.A.S.T.L.E. would dramatically reduce the load on the registrar's office regarding room allocation, and coupled with a timetable allocation system, would allow for a fully automated allocation process.

Other fit parameters for example could include:

- The distance from the professor's department, which makes stability harder to achieve, but improves practical aspects
- Rigorous equipment matching, which leads to losing strategy-proofness, and more unmatched courses, but is necessary for some subjects like computer science
- A long-term fairness parameter, where ties in the capacity would be broken by favoring the course that had the worse room rank in the previous allocation, this would also lose strategy-proofness, but would allow for a better perceived long-term impact.

Overall, the system is very modular and can be easily adapted to the needs of the university, and the preferences of the faculty and students. Its quickness and efficiency would allow for a quick and easy allocation process, and the stability of the algorithm would allow for a long-term deployment.

Next steps for C.A.S.T.L.E. would be to implement the timetable allocation system, using most probably a graph coloring algorithm, and to test the system on a larger scale, with more room types and courses. The algorithm could also be complemented by a front-end interface, where faculty members could input their preferences, and see the results of the allocation. This would allow for a more transparent allocation process, and would allow for the faculty to understand the process better.

CONCLUSION

The Capacity-Aligned Stable Teaching Location Engine (C.A.S.T.L.E.) represents an advancement in addressing the complex challenge of classroom allocation in higher education institutions. Through careful market design principles and algorithmic implementation, it achieves several critical objectives that current manual systems struggle to meet.

The system's core strength lies in its balance of theoretical guarantees with practical applicability. The modified Deferred Acceptance algorithm ensures stability and Pareto efficiency, while maintaining limited manipulability (and thus relative strategy-proofness) from faculty- properties that are essential for a long-term deployment in university settings.

These theoretical foundations are complemented by empirical evidence from simulations using real preference data from Ecole Polytechnique, demonstrating consistent allocation rates above 95% with an average choice rank below 2, even with limited preference expression.

Additionally C.A.S.T.L.E.'s computational structure, achieving $O(C \cdot R)$ average-case complexity, enables real-time allocation adjustments while handling the scale of modern universities. Its modularity allows the administration to customize it with different fit functions, accommodating institution-specific priorities. This flexibility, combined with its ability to maximize resource utilization of highly-demanded rooms, addresses the growing challenges faced by universities, as they expand their program offerings and student populations.

Finally, the room allocation process is transformed from an opaque, manual task into a transparent, automated preference-based system that captures and responds to stakeholder needs. Surveys at Ecole Polytechnique demonstrate preferences exist among faculty and students, and that these can be incorporated into the allocation process while maintaining high efficiency. This represents a significant improvement over current systems, which often rely on single criteria like capacity or require extensive manual coordination.

In the future as universities continue to face pressure on their physical resources or expand, using systems like C.A.S.T.L.E. will prove necessary. While further developments in areas such as timetable integration and user interface design would enhance its practical implementation, already provides a robust theoretical foundation for more efficient and satisfying classroom allocation in higher education institutions.