Optimization of Course-to-Room Assignment to Reduce Travel of the Students

Auteur : Gerard, Manon
Promoteur(s) : Louveaux, Quentin
Faculté : Faculté des Sciences appliquées
Diplôme : Master en ingénieur civil en informatique, à finalité spécialisée en "intelligent systems"
Année académique : 2024-2025
URI/URL : http://hdl.handle.net/2268.2/24933

UNIVERSITE DE LIEGE
FACULTE DES SCIENCES APPLIQUEES

Optimisation of Course-to-Room Assignment to Reduce Travel of the Students

Gerard Manon

Thesis presented to obtain the degree of:
Master of Science in Computer Science and Engineering

Thesis supervisor:
Louveaux Quentin

Academic year: **2024 - 2025**

Acknowledgements

I would like to thank my supervisor, Professor Q. Louveaux, for his support and guidance throughout the development of this thesis. His feedback and help allowed me to complete my work. I am also particularly grateful to him for having taught the courses that provided me with the knowledge to produce this work.

I would also like to thank the SEGI [1], in particular Mrs. C. Erpicum, for providing me access to real-world data. The availability of this data was crucial to the completion of this work.

I am also grateful to Mrs. K. Dethier, the facility manager, for her constructive feedback on the results and for the time she took to adapt the allocation for next year.

Additionally, I would like to thank "Team Montéfiore", composed of L. Boveroux, G. Derval, E. Estève, Q. Louveaux and B. Miftari, for the opportunity to participate in the Integrated Healthcare Timetabling Competition [2]. All the optimisation knowledge they shared with me during the meetings for this competition allowed me to improve in optimisation, which was a great help for this work.

Summary

This thesis addresses the challenge of optimising student travel within a university based on a predetermined timetable. Optimising travel would reduce reliance on cars and buses, leading to less traffic and a more pleasant daily schedule for students. The objective is to minimise the total distance students must travel between consecutive classes, a problem especially pertinent for geographically dispersed campuses like the University of Liège. However, this problem is inherently intractable.

To address this, this research employs an approach to mitigate inter-district movement, penalising longer commutes that are most impactful. An optimised course-to-room assignment model is developed to allocate courses to available rooms, minimising inter-district student travel.

An attempt to identify whether the problem could still be solved optimally by splitting it into sub-blocks of weeks was tested, hoping it could perform quickly. Unfortunately, that was not the case, but the complete model performs in a reasonable time, so this is not a problem.

Résumé

Cette thèse aborde le défi d'optimiser les déplacements des étudiants au sein d'une université en fonction d'un emploi du temps prédéterminé. Optimiser les déplacements réduirait la dépendance aux voitures et au bus, ce qui entrainerait moins de trafic et le plaisir des horaires quotidiens des étudiants. L'objectif est de minimiser la distance totale que les étudiants doivent parcourir entre les classes consécutives, un problème particulièrement pertinent pour des campus géographiquement dispersés comme l'Université de Liège. Cependant, ce problème est insoluble par nature.

Pour remédier à cela, cette recherche utilise une approche pour atténuer les mouvements entre districts, pénalisant ainsi les trajets plus longs qui sont les plus impactants. Ainsi, un modèle optimisé d'attribution de cours aux salles est développé pour attribuer les cours aux salles disponibles, minimisant ainsi les déplacements entre districts des étudiants.

Une tentative d'identifier si le problème pouvait encore être résolu de manière optimale en le divisant en sous-blocs de semaines a été testée, dans l'espoir qu'il puisse fonctionner rapidement. Malheureusement, ce n'était pas le cas, mais le modèle complet fonctionne dans un délai raisonnable, donc ce n'est pas un problème.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

Imagine rushing to the bus stop as soon as your course ends, quickly trying to make it to your next class in Sart-Tilman after a lecture in the city centre of Liège. This is the daily reality for many students at the University of Liège, where courses are often scheduled across its widely spread locations.

Recent mobility surveys at the University of Liège highlight how big the travel issue is. In 2024, 48% of students indicated changing sites during the same day [4]. This number has grown from 41% in 2013 [5] and 45% in 2021. This means nearly half of the students must leave one site to attend classes at another, often with limited time in between. This limited time becomes a stressful race against the clock, often resulting in students arriving late or wasting precious minutes.

These trips are not limited to the movement between the city centre and Sart-Tilman. The 2024 survey data shows that 10% of students report internal movement within the city centre district, while 12% move between buildings in Sart-Tilman. However, the longest distance involves 26% of students who commute between the city centre and Sart-Tilman during the day. It is estimated that student movement between the Sart-Tilman district and the city centre generates 5,851 movements each day. This inter-district travelling is not evenly distributed among students. According to the 2024 mobility data, some faculties contribute disproportionately to these movements. Students in the Faculty of Psychology, Logopedics and Educational Sciences are the most affected. Half of them are travelling between districts, generating an estimated 1,361 daily movements. Similarly,

43% of students in the Faculty of Law, Political Science and Criminology are impacted by this travel, resulting in 1,029 movements. The Faculty of Medicine also shows significant inter-campus travel, with 22% of students involved in approximately 1,166 movements each day. In addition, it is noted that these statistics probably underestimate the amount of travel as these transfers do not necessarily occur only once a day. Many students reported needing to travel multiple times between districts during the same day due to how courses are scheduled [4].

Furthermore, the buses that connect the districts are frequently overcrowded. This forces students to wait for the next bus, which is equally full, or rely on their cars instead. This dependence on cars worsens traffic congestion on the main roads around the university, especially during peak hours. This also has a big environmental impact. More vehicles on the road contribute to more pollution and an increase in carbon footprint.

Moreover, this pressure on public transport is growing. Bus lines connecting the city centre and Sart-Tilman are often full during peak times. According to the 2013–2014 mobility report, just over 10% of students complained about bus saturation. By 2024, this increased to 70%. This is correlated to an increase in bus usage within the ULiège community. It went from approximately 40% of 21,013 students in 2013 to 44% of 24,048 students in 2024, representing about 2,000 additional students depending on buses for their daily commute [4–6].

It is important to note that the public transportation network in Liège has recently undergone a significant transformation with the introduction of the tram and, in a few months, with the electric busway lines [7]. These developments should contribute to reducing bus overcrowding issues and improving the overall environmental impact of commuting. However, even with these improvements, public transport, which is generally more sustainable, still contributes unnecessarily to emissions when students must travel due to inefficient scheduling. Reducing these avoidable trips by better organising class locations could decrease the university's environmental impact. This effort contributes to the eleventh goal of the United Nations Sustainable Development Goals, which encourages more sustainable cities and efficient transport systems [8]. This is done by reducing unnecessary travel and by improving how the university is organised across its different locations, as discussed in the next paragraph.

In addition to optimising room assignment, there are plans to address the root of the problem by relocating some faculties. Specifically, the University of Liège is considering moving the Faculty of Law and the Faculty of Humanities back to the city centre [9, 10]. Different locations, such as the convention centre of Liège, the former Chiroux site and

the Opera, were evaluated. If this were to arrive, it could further reduce the need for students to move between the city centre and the Sart-Tilman campus. This suggests that this problem is more global than room assignments.

The University of Liège's current room assignment process does not consider the impact on student travel. When making room reservations, facility managers focus on respecting the room capacity and requirements provided by teachers. This approach often misses the impact on student movement between classes. Furthermore, room assignments frequently remain unchanged year to year if the course schedule is similar, potentially perpetuating existing inefficiencies in student travel.

The enormous number of possible room allocations makes finding an optimal solution impossible to do manually. Even for computers, exhaustively exploring every single combination would be incredibly challenging. This is precisely where Mixed-Integer Linear Program (MILP) comes in handy. MILP provides a more efficient approach than simply trying every allocation. By using powerful techniques like tree pruning, MILP can intelligently search the large solution space, making it a far more practical tool for complex problems.

1.2 The Course-to-Room Assignment Problem

Given a predetermined timetable, the course-to-room assignment problem addresses the challenge of optimally assigning courses to available rooms. Its objective is to minimise the total distance students must travel between classes. This is pertinent in scenarios where a university campus is geographically spread out, like at the University of Liège. Louveaux introduces this problem, which was presented at the ORBEL conference [11]. This optimisation problem could also be transposed to other problems, like the assignment of goods inside a warehouse. In that case, goods will have to go to storage locations in a way that would reduce the travel of workers picking the goods.

The basic principle of the problem is to improve student flow by ensuring that courses taken consecutively by the same group of students are assigned to rooms that are as close as possible. This must respect a fundamental hard constraint, which is that each course must be assigned to a room with sufficient capacity.

This problem can be formulated as a MILP where an objective function must be minimised while respecting constraints.

The mathematical formulation for the travelling student problem is the following:

Indices

Index	Set	Description
r	$R = \{1, 2, ..., N_R\}$	set of rooms
c	$C = \{1, 2, ..., N_C\}$	set of courses

Table 1.1: Indices and the sets they belong to

Data

Data	Domain	Description
$\text{dist}[r_1, r_2] \in \mathbb{R}$	$\forall r_1, r_2 \in R^2$	distance between rooms r_1 and r_2
$\text{nb\_St}[c_1, c_2] \in \mathbb{Z}$	$\forall c_1, c_2 \in C^2$	number of students sharing courses c_1 and c_2
$\text{small}[c] \subseteq R$	$\forall c \in C$	rooms which are too small for courses c

Table 1.2: Data and its description

Variables

$$\text{assigned}_{c,r} = \begin{cases} 1 & \text{if course } c \text{ is allocated to room } r \\ 0 & \text{otherwise} \end{cases} \qquad \forall c \in C, r \in R$$

Objective: minimise the sum of distances between rooms for all pairs of courses following each other that share students, weighted by the number of students shared

$$\min \sum_{r_1, r_2, c_1, c_2} \text{dist}[r_1, r_2] \cdot \text{nb\_St}[c_1, c_2] \cdot \text{assigned}_{c_1, r_1} \text{assigned}_{c_2, r_2}$$

Constraints

- Each course must be assigned to exactly one room

$$\sum_r \text{assigned}_{c,r} = 1 \qquad \forall c \in C$$

- A course can not be assigned to a room too small

$$\text{assigned}_{c,r} = 0 \qquad \forall c \in C, r \in \text{small}[c]$$

This formulation highlights that the problem can be formulated as a Quadratic Assignment Problem (QAP). It can be expressed as a graph $G = (V, E)$ with distances between two nodes. The problem involves assigning a set of "facilities" to a set of "nodes". Here, the courses serve as facilities to be assigned to the rooms, which are the nodes. The objective of a QAP is to minimise the sum of the flow between two facilities multiplied by the distance between their assigned locations. In our case, the flow between two courses c_1 and c_2 is represented by the number of students sharing them (nb_St$[c_1, c_2]$), and the distance is the distance between the rooms r_1 and r_2 where they are taught (dist$[r_1, r_2]$).

QAP is a well-known NP-hard combinatorial optimisation problem, meaning that there is no known algorithm that can solve it in polynomial time [12]. One common approach to solve quadratic programs is to linearise them by introducing auxiliary binary variables. For this problem, a new binary variable $y_{c_1 c_2 r_1 r_2} = \text{assigned}_{c_1 r_1} \text{assigned}_{c_2 r_2}$ could be defined. However, this approach leads to an explosion in the number of variables. For example, with 300 rooms and 400 courses to schedule daily, introducing $y_{c_1 c_2 r_1 r_2}$ would result in more than a billion new binary variables, making a pure linearization computationally impossible. In fact, QAP problems with more than 30 facilities are hardly ever solved in a reasonable time with current techniques [13]. Due to the hardness of the problem, a simplified formulation must be investigated.

1.3 Objective

The objective of this thesis is to develop and investigate optimisation models aimed at reducing student travel between districts. This thesis assumes a predetermined and fixed course timetable. Although timetable adjustments could contribute to reducing travel, this thesis does not investigate them.

This research focuses on minimising student movement by developing an optimised course-to-room assignment model to assign courses to available rooms. This approach ensures that students' overall travel distances between their various classes are minimised.

By focusing on room allocation, the number of students travelling by bus during the day will reduce, thereby reducing overcrowding. It could also decrease the number of car trips around campus, leading to less traffic and a more manageable daily schedule for students.

1.4 Outline of the thesis

The work is structured into chapters, each addressing a specific aspect of the research.

- **Chapter 2: State of the art**

  This chapter presents the current state of research covering how related problems were addressed in the literature.

- **Chapter 3: Models**

  It begins by defining key terms relevant to the dataset. Then, the chapter presents the mathematical formulations of the models and the algorithm for the room assignment.

- **Chapter 4: Dataset description and processing**

  The real-world dataset is described and the data formatting process is explained.

- **Chapter 5: Weight tuning**

  A methodology to tune the objective function weights is proposed.

- **Chapter 6: Results and discussions**

  The main results are presented through a comparison of the algorithms used.

- **Chapter 7 Conclusions and perspectives**

  It summarises the main findings and contributions of this work and discusses future work.

Chapter 2

State of the art

The problem of reducing travel for students is quite specific to the geographical organisation at ULiège, which is composed of different districts. Although this problem is not studied in the literature, it is related to several optimisation problems. Course timetabling is an important problem all around the world and can be split into a lot of sub-problems [14]:

- **Course timetabling**: The problem of assigning course sessions to time periods, including diverse constraints. Integer programming is used for its formulation, but it is often too hard to solve alone [15]. Therefore, it is frequently combined with meta-heuristics, which are also used independently. These meta-heuristics approaches include tabu search [16], simulated annealing [15, 17], genetic algorithms [18] and ant colony [19, 20]. Another approach is graph colouring, where distinct colours are assigned to courses to represent time slots [21].
- **Class-teacher timetabling for high schools**: From students already allocated to classes and teachers to courses, this problem determines the schedule of class-teacher meetings. Without side constraints, it is solved in polynomial time using a network flow algorithm [22]. Heuristics are also used such as tabu search [23, 24], simulated annealing [25] and genetic algorithms [26];
- **Teacher assignment**: Teachers are assigned to courses in a way that maximises a preference function. It is solved in polynomial time using a network flow algorithm [27]. Logic programming [28] and integer linear programming with variables representing the full schedules [29] are also tested.

- **Student scheduling**: Students are assigned to course sessions when a course is split into groups. This should respect room capacity, balance the size of groups and ensure a conflict-free schedule for students.
- **Classroom assignment**: Each course session is assigned to a room satisfying constraints like the size, location and requirements. This is typically done after having the course timetabling to simplify the overall process, even though this problem alone is still NP-complete [30].

There is no standard solution to solve all of these problems since they must always be adapted to the requirements of each school or university. Therefore, there are no test instances which are used across multiple research studies to have an accurate comparison of the performance of a proposed solution.

It is noted that the literature usually focuses on university programs. However, some research is also done for high school. Indeed, high schools often have predetermined classes and only a few programs. This results in only a teacher assignment and scheduling problem, the class-teacher timetabling. Usually, classroom assignments are not an issue since all rooms tend to have the same capacities. On the other hand, for university, one of the issues is that students have a lot of flexibility in their course selection. This results in courses having diversified capacities and therefore requiring rooms with different capacities.

This thesis focuses on the last two sub-problems, with some key differences. For student scheduling, the group sizes are already determined and students might have conflicts which should be minimised when possible. For classroom assignments, researchers typically consider that the schedules are the same from one week to another, which is not the case at ULiège. In addition to a diverse schedule, the room allocation is also allowed to occasionally change.

2.1 Classroom assignment

Without considering the student's travel, the room assignment problem is quite common. This problem formulation can be applied in many fields, including hotel room assignments, job scheduling on machines with varying capabilities, assigning airport gates to flights, among many others [31–34].

Early studies by Glassey and Mizrach for the University of Berkeley use a discrete Linear Program (LP) for the classroom assignment [35]. It is solved by decomposing it and using

a "hardest problem next" heuristic. This heuristic therefore first assigns classes happening in time slots with the highest ratio of unscheduled classes to available room. For this time slot, it first assigns the largest room-demanding class.

Gosselin and Truchon use an integer LP to solve the room assignment, where rooms are grouped in different preference sets. From this, similar rooms and classes are grouped to reduce the problem size. This reduction of the size is a key factor for being able to solve the LP. In a second phase, a precise decision of which of the rooms is assigned to the class is made [36].

2.2 Student scheduling

Grouping of students can be either random, in which case no optimisation is required, or based on some criteria. Most of the time, the groups are formed by preventing conflicts for students and having a balanced size of group [37]. This splitting can be based on school background and academic performance, as it is often the case at ULiège for language courses. Some universities even take into account various criteria such as religion, parents' income, and average grade [38]. Some searches also take into account the opinion of the students through surveys. Groups can also be dictated by students' project groups, which are split into distinctive rooms. Other courses group students from the same curriculum together, which is the nearest approach to the result obtained in this thesis, which only tries to minimise travel [39].

A first heuristic used to solve student scheduling is a three-phase protocol developed by Laporte and Desroches. In a first phase, students are split and assigned to diverse groups without looking at the room capacities or balancing sections. In the second phase, sections are balanced out. Then the third phase takes into account the room capacities [37].

Simulated annealing is also used. Chown, Cook, and Wilding use it based on group project preferences, while Ciebiera and Mucha use it for time slot preferences. Dowsland also uses it to determine the group as well as the minimum number of groups needed to avoid scheduling conflicts [40–42].

Holmberg's research compares simulated annealing to basic heuristics and tabu search for forming balanced-size groups with balanced aspects, like gender. This approach also minimises how often two students are grouped together. Tabu search performs the best in that context [43].

Sabin and Winter implement a greedy algorithm for which they claim that a feasible solution always appears. Their method is the following. They assign weights to each course and then for each student, they derive a weight based on their course program. Students are processed in their order of "awkwardness" and courses are processed based on their number of groups. This makes sure that the most complicated schedules are dealt with first. Students are assigned to the course group which has the least number of students, while ensuring this does not introduce any conflict. A second pass is then performed for the students who have not been assigned when there are no groups without a conflict. Details about this second pass are not fully provided [44].

Chapter 3

Models

3.1 Concepts

While Section 1.2 simplifies the data by considering it is simply courses which must each be assigned to a room, the data is more complex. Therefore, it is essential to introduce the concepts which highlight how the data is processed and structured.

3.1.1 Courses

Courses take place during two *terms*. The first term, Q1, happens from September to December and the other, Q2, from February to May. In the Belgian higher education system, these terms are in reality followed by exams. A last exam period is also held for students who failed the exams. However, exams are not included in the scope of this work, which focuses only on the assignment of rooms to course lectures. Each term includes at least 12 weeks of learning activities [45], with additional holiday weeks during which no courses occur. The fact that terms are organised into weeks does not mean it is sufficient to focus on a single week and extrapolate the results, as course schedules vary across weeks. These variations are due to teachers having external constraints, or simply the organisation of the course, which might not require it to occur each week. For example, laboratories are often quite rare, so they usually occur in only one or two weeks. Some courses also require the theoretical content to progress to a certain point before exercises or laboratories can be introduced.

Each course is characterised by its name and is associated with a *faculty*. This faculty is determined by the one most students of the course are in, differing from the data's

faculty which indicates the faculty responsible for the course. Our focus on reducing student travel prioritises the students' faculty over the responsible teachers'. Each faculty has a set of preferred zones where the course should ideally be given. These zones are usually close to the faculty buildings since both teachers and students want to avoid travelling far. The degree to which each faculty prefers specific zones will be explained later. The *set of students* enrolled in a course, and therefore its *capacity*, is also known. This makes it possible to determine how many students are shared between different courses. Identifying shared students between courses helps determine potential student travel between locations. Minimising this travel is a key objective.

3.1.2 Sessions

Moreover, courses are divided into *sessions*, which represent the time slots when the course activities happen. These sessions highlight the structure of the course. For example, a course might have a theoretical part in a single room, followed by an exercise part where students are split into two rooms. In such cases where multiple rooms are needed simultaneously, the session is split into several separate sessions, each assigned to a different room, as illustrated in Figure 3.1. It is said that these sessions are *related* to each other. This splitting of sessions also happens for course sessions which are too big to fit in only one amphitheatre. Due to this splitting, the capacity of each session must be based on the *percentage* of students following the course who are assigned to the session.

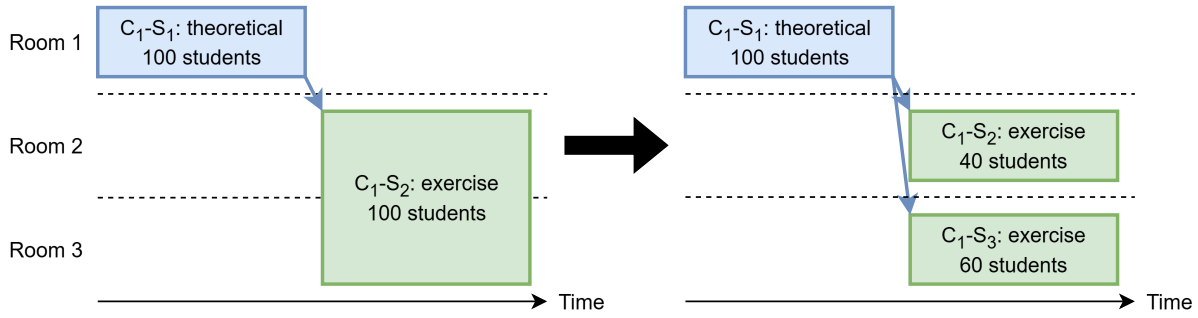

Figure 3.1: Example of a session split due to multiple rooms usage

From the session capacity, it can be deduced the capacity of the room required. However, room capacity requirements can sometimes be dictated by the teaching method needs rather than just student numbers. For instance, the course "INFO0952-1: Additional information technology" uses a room with 565 seats even though it has just under 200 students. This is because the teacher and assistants need to be able to go through the rows to answer the students' questions. Therefore, a room with twice the student capacity is necessary to ensure that every other row remains free.

The information about when and where the sessions take place is available. This includes the day of the week, the start and end times and the weeks the sessions occur. Sessions are often repeated across several weeks, ideally happening in the same location. A course is allowed to have sessions on different days of the week, and these days do not need to be consecutive. They can be sessions every day of the week, excluding Sunday. When courses extend across both terms, their sessions are divided according to the term in which they occur. Each session is also assigned a category, but this information is only slightly used as there are no official definitions. Therefore, faculty members tend to use them in different ways, creating inconsistencies.

3.1.3 Groups

A *group* refers to a subset of students from a course who attend a specific session. This concept becomes relevant when a course is divided into multiple smaller sessions, each accommodating a portion of the total student cohort. These smaller sessions are not necessarily concurrent. They can happen at various times and on different days. While each group has a specific size, the sum of all group sizes within a particular division accounts for the full course enrolment.

Courses can be divided into multiple such sets of groups, typically corresponding to different categories of activities. For instance, one type of activity might require three distinct groups to cover all students, while another activity within the same course might necessitate a larger or smaller number of groups. Thus, each distinct division constitutes a *set of groups*, ensuring that the full student cohort participates in each specific activity type.

3.1.4 Rooms

Each course session must be assigned to one *room*, which can only have one session at a time to avoid scheduling conflicts. For each week in which the session takes place, a room is assigned. The weekly room allocations from the ULiège schedule are referred to as the *past* rooms. The new room allocation generated by the optimisation program is called the *assigned* rooms. Since the room assigned to a session may vary across weeks, a *privileged* room is defined as the one most frequently used among the past rooms, and similarly, a *new privileged* room for the assigned rooms. The idea of this privileged room is to have a main room in which a session should be the majority of the time, for the quality of the schedule. Since the data did not require sessions to be assigned to the same room across the week, some sessions can originally have no privileged room. Nevertheless, the original assignment will change to introduce a new privileged room when optimising.

Rooms are characterised by their name, type and capacity. The capacity is quite an important factor since every student must be able to fit in the room. These rooms are situated in a *zone* of the University. Indeed, at the University of Liège, buildings are grouped into several zones, usually linked to a field of study. These zones are then grouped into the *district* of the Sart-Tilman or the city centre. Maps of these districts and their zones are available in Appendix A.

The complexity of room allocation at the University of Liège extends beyond courses. ULiège serves as a hub for a wide range of academic, professional, and public activities. Therefore, the university's rooms are not used exclusively for student courses but also for a diverse range of activities, such as conferences and meetings. In addition, the university sometimes needs to renovate its rooms or at least do some maintenance in them, so they might not be available for courses. All these factors are taken into account by including the room's *unavailability* for teaching.

The relationships between all the concepts introduced are summarised in the UML diagram in Figure 3.2.

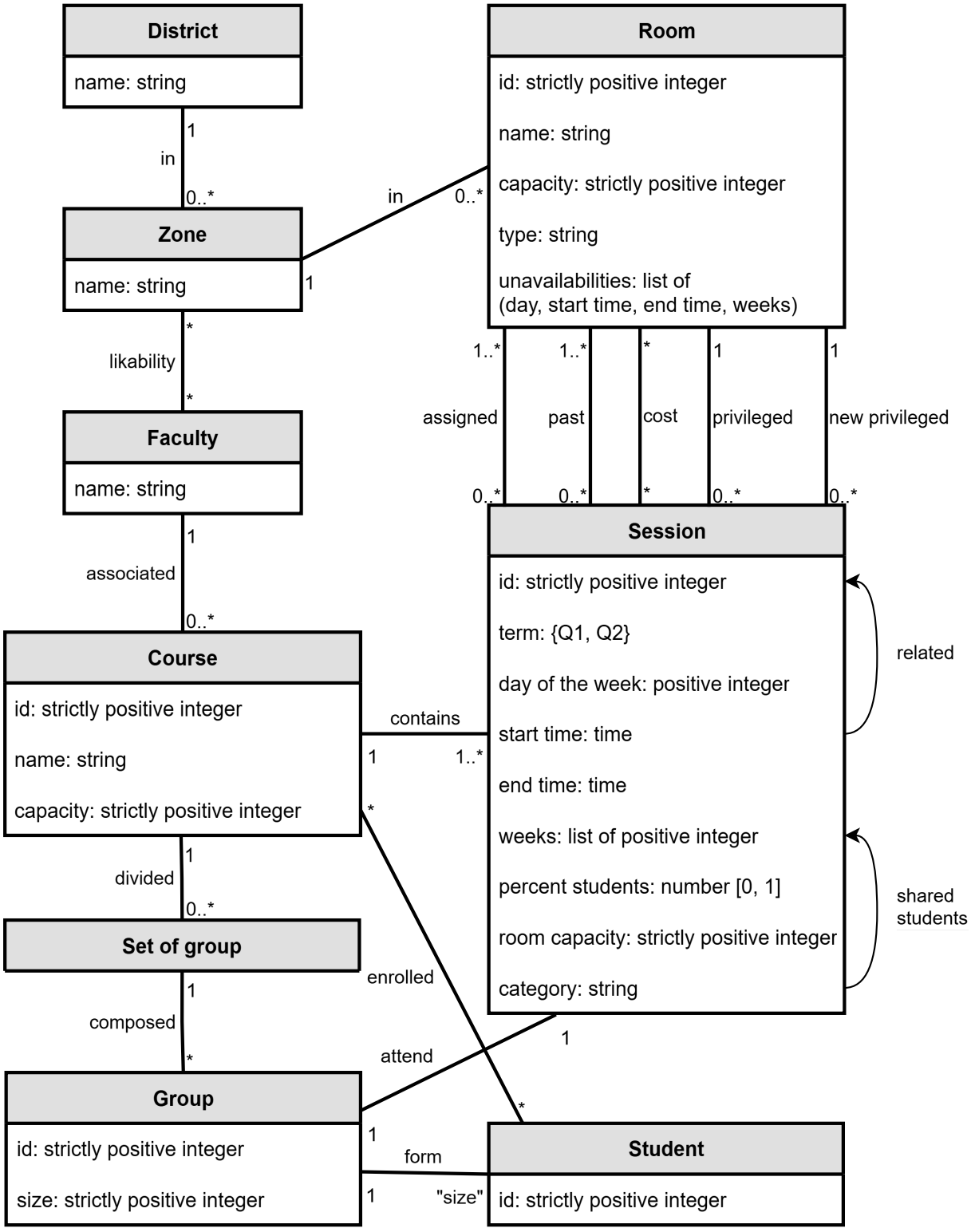

Figure 3.2: UML diagram representing the relationships between courses, sessions, rooms, and students.

3.2 Course-to-Room assignment models

3.2.1 Simplified model

Since the course-to-room assignment problem introduced in Section 1.2 is intractable, it must be simplified. Instead of reducing the distance between every room, Louveaux proposes a linear model that focuses on limiting the longer distances between rooms which are in different districts [11]. The main idea is to penalise situations where a student has to change district during the day. This choice of focusing only on the inter-district travel seems reasonable since it is the most impactful. Travel inside each district is less of a problem and it is considered to be possible on foot in a reasonable amount of time.

This formulation does not yet introduce the concept of sessions instead of courses, which could be held more than once. It is formulated for a representative "typical" week. However, as mentioned previously, course schedules vary across weeks.

The simplified model introduces an objective coefficient $\text{cost}[c, r]$ that reflects the cost associated with assigning a course c to a room r based on its district. Additionally, a penalty is applied in the objective function for any change of district for pairs of courses that share at least 5 students.

$$\min W_1 \sum_{c,r} \text{cost}[c, r] \cdot \text{assigned}_{c,r} + W_2 \sum_{c_1,c_2} \text{nb\_St}[c_1, c_2] \cdot \text{pen}_{c_1,c_2}$$

The problem now becomes a multi-objective optimisation problem. There are several ways to deal with this, but a weighted cost function approach is used as it is a practical way to come back to a single objective. This requires assigning a weight to each objective, which indicates its importance. These objectives are then summed to obtain a single objective. Choosing the weights W_1 and W_2 is therefore an important aspect which is studied in Chapter 5.

Let D be the set of all districts, and for any district $d \in D$, let $R[d]$ be the set of all rooms located in d. The penalty constraint for a change of district for a pair of courses c_1 and c_2 between two different districts, d_1 and d_2, can be formulated as:

$$\text{pen}_{c_1,c_2} \geq \sum_{r_1 \in R[d_1]} \text{assigned}_{c_1,r_1} + \sum_{r_2 \in R[d_2]} \text{assigned}_{c_2,r_2} - 1 \qquad \forall c_1, c_2 \in C^2, d_1, d_2 \in D^2, d_1 \neq d_2$$

The introduction of this type of constraint allows for a completely linear binary formulation, making the problem tractable for decent-sized instances.

3.2.2 Extended problem

The real problem does not simply consist in assigning a room to each course. As stated in Section 3.1, instead of courses, sessions should be used since courses can be composed of multiple sessions. In addition, it is not sufficient to focus on a single week's assignment due to changes in the schedule. This assignment must be done for every week where courses can take place. Since travel only counts when it happens during the same day, there is independence between days. Therefore, the problem can be solved by considering one day at a time. This simplifies the model to solve, creating six smaller models to solve. In addition, terms are considered to be independent from one another, so once again the models can be split between the two terms. This leads us to twelve models which each treats data from one specific day of the week, of a term. It should therefore be optimised for each day when courses can be held and for each term.

The previous problem does not take into account the current session allocation provided by the university to create a new optimal one. However, this is not what is wanted. The session should stay quite similar to how it is currently affected. For this, extra constraints must be added to allow a maximum number of changes of privileged rooms. There are multiple reasons to keep room allocations similar to the one provided by the university. The primary one being that the model does not account for all real-world constraints, as described in Section 3.2.3. By limiting changes, the risk of generating a schedule that violates these unrepresented constraints is reduced. Secondly, faculty managers want control of this allocation, which is therefore done manually, while keeping a lot of allocations similar to the previous year. If all courses were to change, this would add up to too much work to verify if the allocation is possible and make decisions. Having only a model that suggests small changes to an existing schedule can be used by the university since it allows for a manual verification and decision, even when some constraints are not represented. Moreover, sessions should tend to be assigned to the same room across the weeks for the quality of the schedule.

In addition, only travel close in time is considered. In this case, when there is less than 2 hours between courses. This is due to the fact that when there is more time it is considered that students will have the time to change districts without suffering from inconveniences like the traffic or bus congestion. This is a choice since it could also be defended that these travels still contribute to bus occupancy and traffic, so they should also be minimised. In addition, travel is considered between sessions when there are at least 5 students shared between their courses.

For the model, sessions happening at the same moment must be identified to make sure the rooms are not occupied by two of them. In order to simplify the problem, this is computed for 15-minutes time slots. This 15-minutes slot is quite realistic with the data since usually courses start and end at time "multiple" of 15. However, there are exceptions for sessions. In this case, the time slot will artificially extend the time that the session occupies the room. This could create some issues as illustrated in Figure 3.3. In fact, since courses are extended, it can result in sessions that cannot happen in the same room, even though nothing prevents it based on the actual session times.

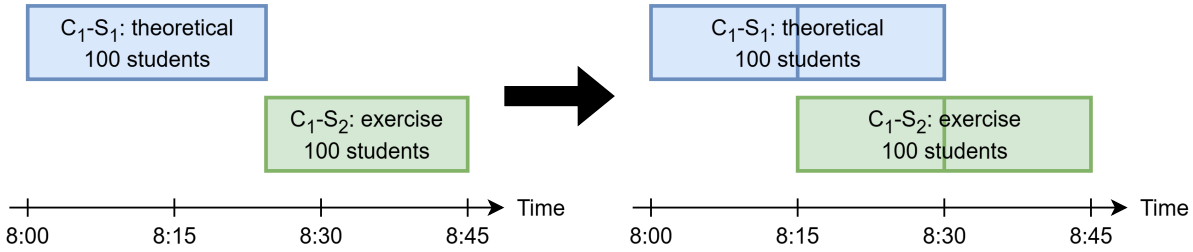

Figure 3.3: Example of artificial conflict created when the sessions' schedules are extended to fit 15-minute slots

Furthermore, a *cost* is computed for each session-room pair to indicate how suitable a room is for a given session. This cost is based on two main factors. The first one is the compatibility between the room's capacity and the session's required capacity. The second one is based on the likeability between the faculty of the sessions and the zone of the room.

For capacity, a simple linear cost is applied, but with different slopes depending on whether the room is larger or smaller than required. If the room is big enough, the cost is based on the excess capacity, with a gentle slope that accepts a larger room more easily. However, if the room is too small, the cost is based on the capacity difference, with a steeper slope. This steeper slope ensures a greater penalty for rooms that cannot contain every student. In addition, when a room has more than a 15% relative capacity deficit compared to what the session requires, an additional penalty is added. This further discourages the choice of rooms which are too small, since it will be impossible for all students to fit in. These rooms are also often directly rejected if they have not been used for the session. This reduces the problem size, while avoiding a solution which would use a room too small, which would be rejected by the faculty manager. The second factor, the room's location, assigns a higher penalty if the room is not in a zone near the faculty.

The extended problem of course-to-room assignment for a day in a term is formulated as follows. This data is computed offline in order to minimise the time required to create and solve the model. The size of the data reported in Table 3.1 corresponds to the first term of the year 2024-2025 on Monday.

Parameters

- max_priv_changes: Maximum number of changed of privileged room
- min_shared = 5: Minimum number of shared students considered
- min_apart = 120: Maximum number of minutes between sessions to take into account the travel of students

Indices

Index	Set	Size	Description
r	$R = \{1, 2, ..., N_R\}$	$N_R = 253$	set of rooms
s	$S = \{1, 2, ..., N_S\}$	$N_S = 1018$	set of sessions
w	$W = \{1, 2, ..., N_W\}$	$N_W = 13$	set of weeks
t	$T = \{1, 2, ..., N_T\}$	$N_T = 48$	set of 15 minutes time periods
d	$D = \{1, 2, ..., N_D\}$	$N_D = 2$	set of districts

Table 3.1: Indices and the sets they belong to

Data

Data	Domain	Description
$\mathrm{W}[s] \subseteq W$	$\forall s \in S$	weeks during which session s is organised
$\mathrm{unav\_R}[s] \subseteq R \times \mathcal{P}(W)$	$\forall s \in S$	set of $(r, \bar{W})$ representing room r is unavailable for session s on weeks $\bar{W}$
$\mathrm{no\_priv\_S} \subset S$		sessions which did not contain a privileged room originally
$\mathrm{priv\_R}[s] \in R$	$\forall s \in S \setminus \mathrm{no\_priv\_S}$	privileged room assigned to session s
$\mathrm{R}[d] \subset R$	$\forall d \in D$	rooms located in the district d
$\mathrm{concurrent\_S}[w, t] \subset S$	$\forall w \in W, t \in T$ s.t. $\lvert\mathrm{concurrent\_S}[w, t]\rvert > 1$[1]	sessions happening on week w during time period t
$\mathrm{shared\_St} \subseteq S^2 \times W$		set of (s_1, s_2, w) for pairs of sessions s_1, s_2 happening on week w sharing at least min_shared students

Data	Domain	Description
$\text{nb\_St}[s_1, s_2] \in \mathbb{Z}$	$\forall s_1, s_2 \in S^2$ s.t. $\leq$ min_apart minutes apart	number of shared students ($\geq$ min_shared) between sessions s_1, s_2
$\text{cost}[s, r] \in \mathbb{R}$	$\forall s \in S, r \in R$	cost of assigning a session s to a room r based on their capacity compatibility and the likeability between the session's faculty and the room's zone.
$\text{small}[s] \subseteq R$	$\forall s \in S$	rooms which are too small for session s, except if it was used in the data

Table 3.2: Data and its description

Variables

$$\text{assigned}_{s,r,w} = \begin{cases} 1 & \text{if session } s \text{ is assigned to room } r \text{ on week } w \\ 0 & \text{otherwise} \end{cases}$$

$\forall s \in S, r \in R, w \in \text{W}[s]$

$$\text{priv}_{s,r} = \begin{cases} 1 & \text{if room } r \text{ is the privileged room assigned to session } s \\ 0 & \text{otherwise} \end{cases} \qquad \forall s \in S, r \in R$$

$$\text{pen\_non\_priv}_{s,w} = \begin{cases} 1 & \text{if a penalty is required for assigning session } s \\ & \text{to a non-privileged room on week } w \\ 0 & \text{otherwise} \end{cases}$$

$\forall s \in S, w \in \text{W}[s]$

$$\text{new\_priv}_{s} = \begin{cases} 1 & \text{if the privileged room assigned to session } s \\ & \text{changes from the one in Celcat} \\ 0 & \text{otherwise} \end{cases} \qquad \forall s \in S$$

$$\text{pen\_district}_{s_1,s_2,w}{}^{2} = \begin{cases} 1 & \text{if a penalty is required for a change of district} \\ & \text{for students following sessions } s_1 \text{ and } s_2 \text{ on week } w \\ 0 & \text{otherwise} \end{cases}$$

$\forall (s_1, s_2, w) \in \text{shared\_St}$

[1] Only time periods where at least 2 sessions are happening that week are considered to avoid introducing useless constraints later on.

Constraints

Constraints are divided into hard (H) and soft (S) types. Hard constraints must be respected. For soft constraints, their violation is taken into account in the objective function, which must be minimised.

- H1: Each session should be assigned to one room each week

$$\sum_{r \in R} \text{assigned}_{s,r,w} = 1 \qquad \forall s \in S, w \in \text{W}[s]$$

- H2: Only one session can be assigned to a room at the same time period

$$\sum_{s \in \bar{S}} \text{assigned}_{s,r,w} \leq 1 \qquad \forall w \in W, t \in T, \bar{S} \in \text{concurrent}[w,t], r \in R$$

- H3: Some rooms are unavailable for some sessions due to events like maintenance or conferences

$$\sum_{w \in \bar{W}} \text{assigned}_{s,r,w} = 0 \qquad \forall s \in S, (r, \bar{W}) \in \text{unav\_R}[s]$$

- H4: Limit the number of changes of privileged room

$$\text{new\_priv}[s] = 1 \qquad \forall s \in \text{no\_priv\_S}$$

$$\text{new\_priv}[s] \geq 1 - \text{priv}_{s,\text{priv\_R}[s]} \qquad \forall s \in S \setminus \text{no\_priv\_S}$$

$$\sum_{s \in S} \text{new\_priv}[s] \leq \text{max\_priv\_changes} + |\text{nb\_no\_priv}|$$

- H5: A session can not be assigned to a room too small, except if in the original schedule it was

$$\text{assigned}_{s,r,w} = 0 \qquad \forall s \in S, r \in \text{small}[s], w \in W[s]$$

- S1: Define a privileged room and a penalty when a session is not happening in the privileged room

$$\sum_{r \in R} \text{priv}[s,r] = 1 \qquad \forall s \in S$$

$$\sum_{w \in \text{W}[s]} \text{assigned}_{s,r,w} \geq \left\lceil \frac{|\text{W}[s]|}{2} \right\rceil \text{priv}_{s,r} \qquad \forall s \in S, r \in R$$

$$\text{pen\_non\_priv}[s,w] \geq \text{priv}_{s,r} - \text{assigned}_{s,r,w} \qquad \forall s \in S, w \in \text{W}[s], r \in R$$

[2] shared_St is quite sparse, so to speed up the creation of pen_district, instead of defining it only for the (s_1, s_2, w) existing, a mapping from 1 to its size was created.

- S2: Minimise travel between courses with at least min_shared students in common and in nb_minutes_apart

$$\text{pen\_district}_{s_1,s_2,w} \geq \sum_{r_1 \in R[d_1]} \text{assigned}_{s_1,r_1,w} + \sum_{r_2 \in R[d_2]} \text{assigned}_{s_2,r_2,w} - 1$$

$$\forall (s_1, s_2, w) \in \text{shared\_St}, d_1, d_2 \in D^2, d_1 \neq d_2$$

Objective

$$\begin{aligned}
\min \quad & \sum_{s,r,w \in W[s]} \text{cost}[s,r] \cdot \text{assigned}_{s,r,w} \\
& + W_{\text{non_priv}} \sum_{s,w \in W[s]} \text{pen_non_priv}_{s,w} \\
& + W_{\text{district}} \left[\sum_{s_1,s_2,w \in \text{shared_St}} \text{nb_St}[s_1,s_2] \cdot \text{pen_district}_{s_1,s_2,w} \right. \\
& \left. + \sum_{s,w \in W[s],d} \text{add_travel_pen}[w,d] \sum_{r \in R[d]} \text{assigned}_{s,r,w} \right]
\end{aligned}$$

3.2.2.1 Two-phase resolution algorithm

An algorithm was developed to solve the previous MILP faster. The resolution uses a two-phase methodology, which breaks the problem into smaller, more manageable sub-problems. The overall computation times will be reduced while ensuring a solution near the optimal one.

The first phase identifies sessions which room cannot be changed to reduce travel. This is achieved by dividing the entire term into smaller, consecutive blocks of weeks. For each of these blocks, the MILP is run independently. The model allows a maximum number of changes of privileged room B_{max} higher than for all weeks of the term T_{max}. This allows to identify more sessions which could have an impact on travel.

Once the algorithm has run for every block of weeks, the new privileged room is determined. Then, any session that kept its original room assignment is considered as fixed for the next phase. The reason is that if a session's room could have been changed to a room in another district reducing the overall travel, it would have been, since more changes of privileged rooms are allowed. The fact that the assignment remained unchanged implies that the original room was already good for that session, or at least that making other changes reduced more the travel.

The second phase of the algorithm uses the MILP to optimise the remaining room assignments over all weeks of the term. Since a lot of the sessions are already fixed, a smaller problem than the original is left to solve, making the process much faster.

By ensuring the maximum number of changes for the individual blocks (B_{max}) is sufficiently large relative to the overall term's maximum (T_{max}), fixed sessions are assumed to be placed optimally. This allows the overall solution to approach the global optimum, even though it was not solved as a single problem. A small B_{max} can compromise the solution's feasibility and quality. If too many sessions are fixed in the initial phase with a limited number of allowed changes, a suitable room may not be available for a session needing one. This is because the original data does not have privileged rooms, while the model requires a designated privileged room for each. Since the original data does not have a notion of privileged rooms, sessions without one must be able to change their room assignments to acquire one. This flexibility is restricted by a small B_{max}, which can potentially lead to an infeasible problem.

The pseudo code of this resolution is provided in Algorithm 1.

3.2.3 Hypotheses and Limitations

A first limitation of this model arises from the exclusion of professor scheduling constraints. While the algorithm efficiently reassigns courses to different locations based on factors such as room availability and student enrolment, it operates without explicit knowledge of individual professor availability. Consequently, any automated reassignment requires a subsequent manual verification process to ensure that the assigned professor can reach the new location on time. Integrating the professor schedules into the algorithm would represent a significant enhancement, allowing for a more autonomous and conflict-free scheduling process. However, this is not possible since there is no real data about what teachers are doing at a specific time. The only data available could be which sessions it is responsible of. This is not enough since it does not guarantee that the teacher is present at that session, and also prior to the lessons, it is not known if the teacher can meet any of the districts or if they are situated near one, which allows them to be on time. In addition, the teacher might have imperatives after the session.

Another limitation is the presence of unrepresented constraints that affect which rooms a session can be assigned to. These can come from a wide range of factors, such as handicap accessibility requirements, specific room layouts like the number of blackboards, or the need for specialised equipment. For instance, a biological science course might require access to specific bones for tutorial sessions. The sessions must be allocated to

Algorithm 1 Two-Phase Optimisation for Course-to-Room Assignment

Parameters: blocks of weeks week_blocks, maximum privileged changes for blocks of weeks B_{max} and for all weeks of a term T_{max}

Output: Optimised room assignments for all sessions.

```
for each term in terms do
    for each day in days do
        ▷ Phase 1: Room Assignment for each block of weeks
        ▷ The goal is to find room assignments that haven't changed from data.
        for each weeks in week_blocks do
            assignments ← OPTIMISE(sessions for weeks, B_max)
        end for
        for each session in sessions do
            session["new_priv_room"] ← most frequent room assigned in Phase 1
        end for

        ▷ Identify and fix sessions with unchanged room assignment after Phase 1.
        fixed_sessions ← UNCHANGED_ASSIGNMENT(sessions)

        ▷ Phase 2: Final room assignment for remaining sessions
        unfixed_sessions ← sessions \ fixed_sessions
        changed_assignments ← OPTIMIZE(unfixed_sessions, T_max)
        assignments ← SAVE(changed_assignments)
    end for
end for

return assignments
```

a room close to the university's specimen collection, as these materials cannot be easily transported.

The data fails to represent the physical or technical infrastructure connecting specific rooms. For instance, when a large lecture is split into two sessions due to capacity, with one session being broadcasted to a second, smaller room, the algorithm knows the sessions are related. However, it is unaware of which specific pairs of rooms have the necessary network connections to enable this broadcasting functionality. This can lead to an invalid assignment where related sessions are placed in rooms that cannot be linked.

The model also assumes that rooms are free whenever there is no course or planned activity taken into account in the rooms unavailability. In reality, the university uses its rooms for a wide range of external activities, some of which are scheduled after course assignments are made. While the model accounts for planned room unavailability, it does not have data on which rooms are preferred for these external activities. This can lead to a suboptimal solution where rooms are allocated for courses, reducing their availability for future external events.

Finally, the optimisation objective focuses on minimising the overall number of student travels. This approach may not be entirely fair, as some faculties might experience a high number of room changes and travel penalties, while others remain largely unaffected.

Chapter 4

Dataset description and processing

4.1 Dataset description

The data comes from the SEGI [1] and is extracted from the CELCAT [46] calendar, which provides information about existing rooms, courses, their assigned rooms and enrolled students. This information is available in XLSX files. The relevant files for the academic years 2023-2024, 2024-2025 and 2025-2026 were obtained.

In the data provided, what is referred to as a session differs from the use in this thesis. The session here concerns only one specific date at which a room is booked. These sessions miss the notion of repetitiveness across the weeks. In addition, some courses have been split into groups. This means that students following a course are sometimes already split into different groups, so it is known in advance which students attend which session. These groups differ from the notion of group introduced earlier. These groups do not belong to one unique room each time but can require to be subdivided again.

Here is a summary of the files received and their main content:

- Students[1]: lists the students enrolled in each course or each course group;
- Rooms capacity: contains details for each room, including its name, capacity and location;
- Rooms sessions: provides room allocation details, such as start and end times and the type of activity. Information to link each session to a course is also available;

[1]Enrolment data for the 2025-2026 academic year is not yet available when this thesis is written.

- Courses faculty: associates each course with its responsible faculty;
- Courses sessions: contains every course session and information to find the assigned room;
- Groups sessions: similar to "Courses sessions", but with students following a course sometimes already split into different groups.

4.2 Data processing

An important initial task is to structure the raw data to introduce the concepts explained in Section 3.1. To achieve this, the data files are preprocessed into a single JSON file, keeping only what is relevant. The structure of this file is detailed in Appendix B. JSON format, in addition to being a standard to communicate data, has the advantage of being human-readable. This processing serves as the first stage of a pipeline. The generated JSON file is then used as input for the room allocation and group partition algorithms. Although this initial computation step takes some time, its creation as a single pretreated file is a key element for efficiency. By having it independent from the main algorithms, the system avoids recomputing the data every time an allocation is optimised. Overall, this saves time in the whole process execution. This initial preprocessing step is also designed to be independent of the core model to ensure the reusability of the solution developed. This means the system can be adapted for other institutions, provided that a JSON file respecting the specified format is provided. The following explains the main steps of how this file is created from the ULiège data.

For this work, only what is related to Liège is kept, so everything happening in the Arlon and Gembloux campuses is ignored. Sessions happening outside the teaching terms are also filtered out.

Since for courses the faculty required by the program is the one of the students and the teacher responsible, this has to be computed. This is done by trying to identify the faculty of each student, which is identified by keeping the faculty the most responsible of its courses. A student mainly follows courses from its faculty to the exception of specific external courses, like language courses, which could therefore be ignored when taking into account the most present faculty. This makes the hypothesis that a student is enrolled in only one program, which represents most of the situations.

To prevent double booking, a room can typically only be assigned to one session at a time. However, there are cases where sessions from different courses intentionally share a room simultaneously. This often occurs during exercise sessions when a teacher groups

students from several of their courses together. This arrangement allows the teacher and their assistants to be available to all students at once, which might not be feasible with multiple separate exercise sessions. This case also covers a specific organisation of the University of Liège where courses can be split into different *partim*, each treated as a unique course. When several partim have the same content, they might be taught together into a single lesson. These shared lessons are represented by a single session linked to a new artificial course that combines the different courses. This artificial new course now respects the fact that a room can only be occupied by one session from one course at a time. This combination of courses is illustrated in Figure 4.1.

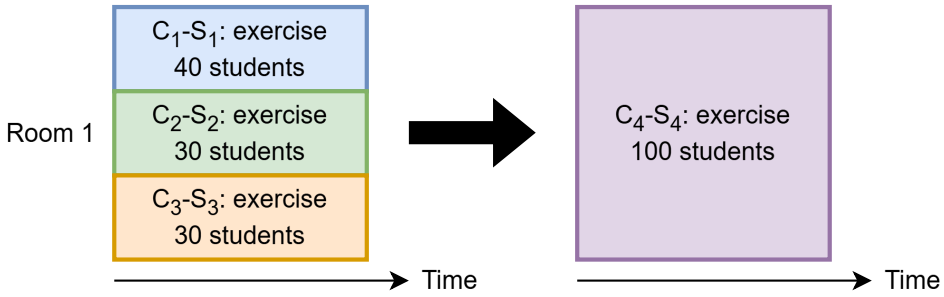

Figure 4.1: Example of sessions from three courses combined into a single session of an artificial course to avoid double room booking

For this thesis, only certain session categories are considered reallocatable, as others are too difficult to move. For example, laboratory sessions often require specific equipment found in only one specific room. Therefore, these sessions can not move from the defined room, so there is no point trying to change that. These sessions are therefore considered fixed and will be included in the room unavailability schedule. However, they are still accounted for the computation of student travel penalties. This represents the additional travel penalty, which is based on the number of students shared with a session that would require travelling if the session were assigned to a specific district. It may seem useless to remove these fixed sessions instead of simply fixing their room assignments. This approach significantly reduces the model's size when passed to the Gurobi solver. Although the solver can easily recognise and handle fixed variables, a large model can be slow to transmit, so reducing its size is important to reduce the overall computation time.

Working with real-world data often presents anomalies. For instance, rooms are sometimes double-booked for a regular course session and a conference or written exam. In these specific cases, since the model does not consider the optimisation of conference allocations, it is decided that the room will be treated as free during the course session. This is achieved by artificially adapting the room reservation schedule for the conference.

4.3 Hypothesis/Limitations

To conduct this work, certain hypotheses are made. A key assumption is the independence between sessions of the same course that occur on different days of the week, as well as the independence between sessions happening in different academic terms. The data provided is not enough to be able to identify if rooms needed to be the same between terms or if it is only a coincidence.

Another assumption relates to student attendance. When calculating student travel, it is considered that all students attend their course sessions. While this does not reflect reality, as some students choose not to attend, the precise data is unavailable. Furthermore, the objective is to create a schedule that provides every student with the opportunity to attend all courses, and therefore, no students are ignored in the travel calculations.

A limitation of the current approach arises from the way courses with multiple simultaneous sessions are handled. While the shared student population across these sessions is identified, the data does not provide information on how students are distributed among the individual rooms. Therefore, a uniform distribution of students is assumed across these parallel sessions. This is a simplification, since in reality it is often non-uniform due to how students are grouped. This work addresses this limitation by introducing a method to create optimal student subgroups to enable a more precise calculation of travel.

4.3.1 Data analyses

This section analyses the key characteristics of the data used for the course-to-room optimisation problem. This analysis is crucial for understanding the problem's scope and identifying critical factors that influence the solution. The insights gained help in several ways: we can assess the problem's complexity, identify critical data-driven constraints like the scarcity of large rooms, and verify that the problem's scale is manageable. The following table and figures are all based on the data from the academic year 2024-2025 at ULiège. The size of the data that will be optimised can be summarised in the following table:

Statistic	**Q1**	**Q2**
Rooms	253	
Courses	3203	
	1877	1596
Sessions	5580	4119

Table 4.1: Statistics about the 2024-2025 data

The number of students per course and per session varies significantly, as shown in Figure 4.2 and Figure 4.3. This is primarily due to the university's curriculum structure, where introductory courses have a large number of students while specialised courses in the more advanced years have smaller groups. This can also be explained by the fact that not every faculty has curricula with a lot of students following it. This variability can have a drawback, which is that it creates highly complex schedules, which therefore often results in student being required to move from one place to another to catch their courses since everybody can't be satisfied. This variability also has a positive aspect, as there are more small courses and sessions, which align well with the number of small rooms available, as illustrated in Figure 4.4. A key challenge lies in the limited number of large amphitheatres, which are almost always booked and schedules must be adapted based on them to allow student to have a place in each of their course session. This limited number and limited repartition over the district force students to travel from one district to another since there are no other rooms available for the course.

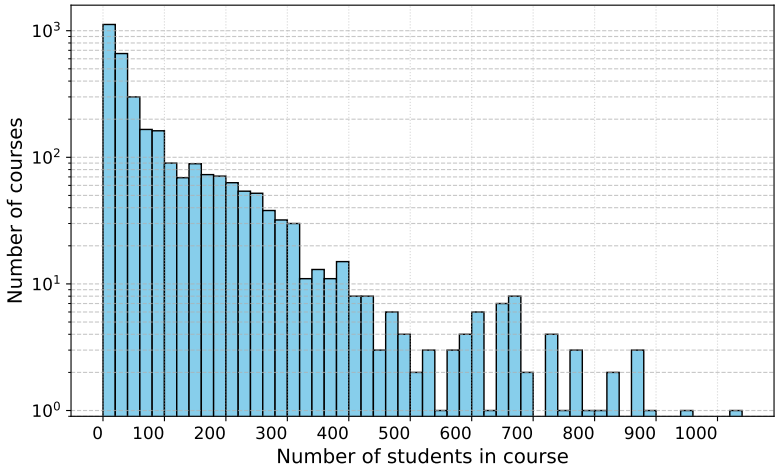

Figure 4.2: Distribution of courses by number of students

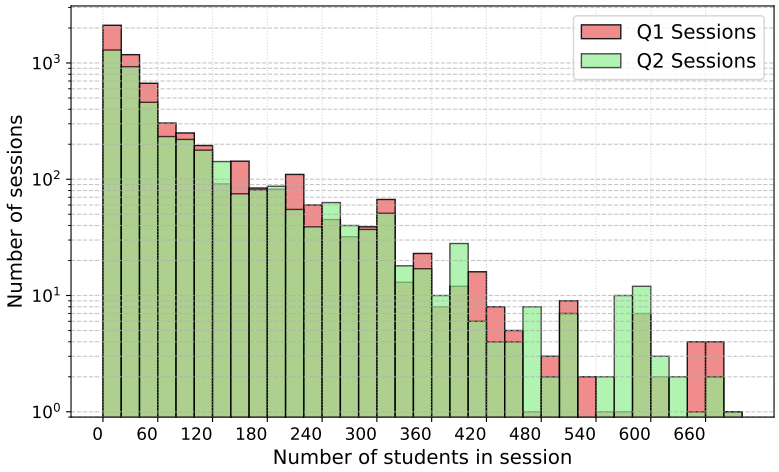

Figure 4.3: Distribution of sessions by number of students

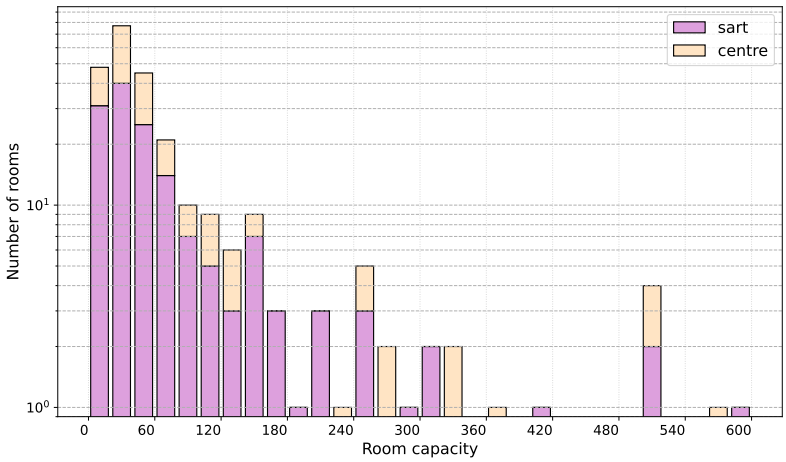

Figure 4.4: Distribution of rooms capacity by districts

The number of sessions by week is presented in Figure 4.5. These range from 1000 to over 2000. The main variability occurs at the beginning and end of each term. This happens because in the first week, there are welcoming sessions which replace course sessions. For the last week, it is usually a result of courses having already finished their programs. Overall, sessions stay in the same order of magnitude from one week to another, excluding the holidays.

Figure 4.6 represents the distribution of the number of students travelling over each day of each term. This only takes into account courses which are considered for the course-to-room assignment. Therefore, this does not include travel between two laboratories as they are both fixed, so the model can not change anything to that. But it takes into account going from a theoretical session to a lab session, meaning that since the laboratory is in

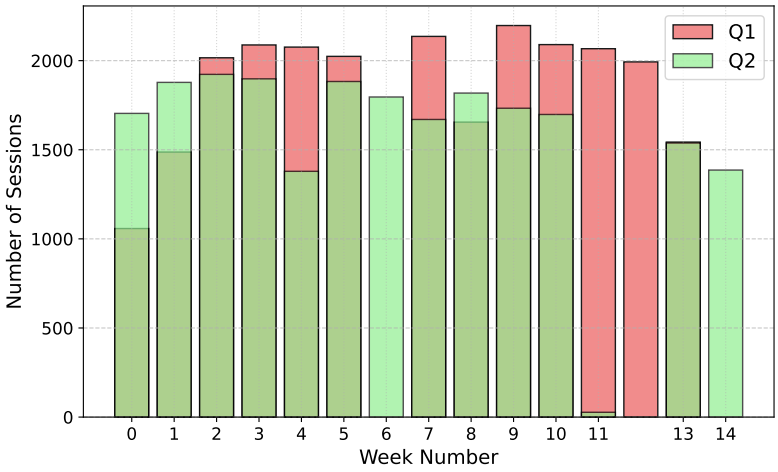

Figure 4.5: Distribution of number of sessions by week

a fixed room, it might be intelligent to assign the theoretical session nearby, or at least in the same district. This data can not directly be compared to the survey introduced in Section 1.1, since a lot of courses are omitted. It can be seen that there is almost no travel on Sunday. This is because there are practically no courses happening that day. It can be seen in Figure 4.7 that there are only a few sessions. Due to all this, it might actually not be the most pertinent day to dig into for reducing travel since the other days are way more important and will have a bigger impact. Figure 4.7 illustrates that the number of sessions per day of the week is quite constant from one week to another, as well as from one day to the other, with the exception of Saturdays.

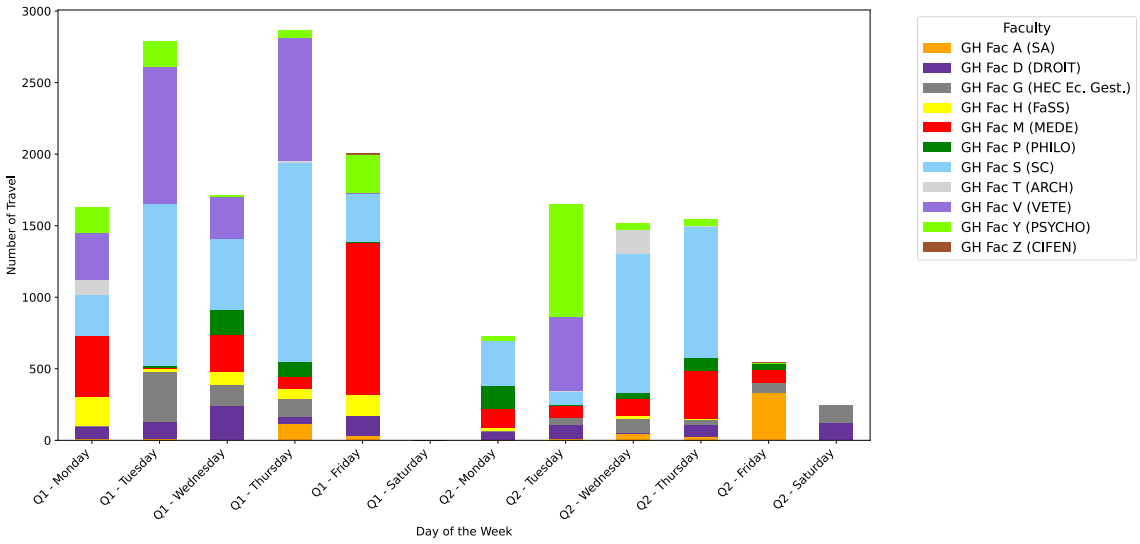

Figure 4.6: Distribution of travel among the faculty by day of the week

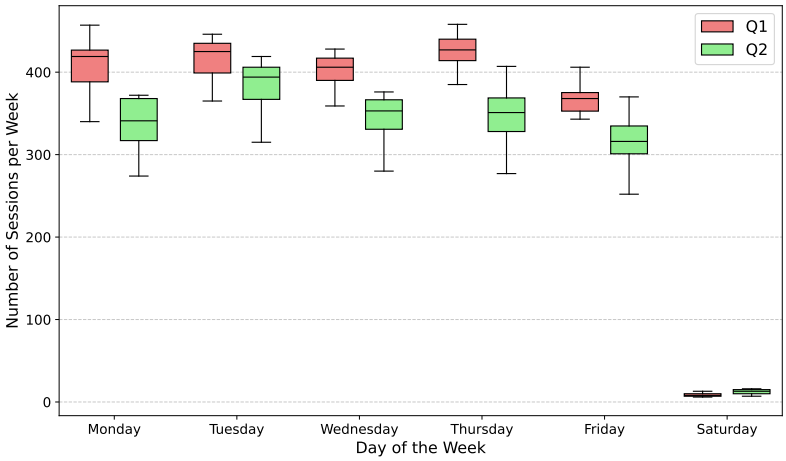

Figure 4.7: Distribution of sessions by day of the week for each term

Overall, days from Monday to Friday seem to be organised quite the same way. Therefore, it is reasonable to use a common approach to optimise the allocation of rooms for each day. A significant challenge, however, is the university's limited number of large rooms. This is a major factor contributing to student travel, as it often forces large courses to be scheduled in rooms that are far from other academic activities. Travelling due to a lack of large rooms is way more impactful than if it were for small rooms, since they do not contain the same number of students.

Chapter 5

Weight tuning

Multi-objective optimisation is a critical technique for solving problems with multiple, often conflicting, goals. Unlike single-objective optimisation, which yields one optimal solution, this approach results in a set of Pareto optimal, or non-dominated, solutions. This set is also known as the Pareto front. A solution is considered Pareto optimal if no objective can be improved without making at least one other objective worse. The Pareto front represents the best possible trade-offs between these competing objectives. Since the problem's objectives were not precisely defined, a weight combination choice is required to make sure too completely define the problem and have a unique solution each time. This unique solution can then be compared with other methods.

A Pareto front is illustrated in Figure 5.1 for a minimisation problem of two objectives f_1 and f_2. Point C is dominated by both A and by B since they both have a lower value in at least one objective without being worse in the other. C is therefore not in the Pareto front. Points A and B are non-dominated and present in the Pareto front. For the course-to-room assignment problem, this front illustrates the balance between minimising student travel, ensuring appropriate room capacity, and respecting privileged room assignments.

Despite the existence of multiple Pareto optimal solutions, a single most preferred solution must ultimately be chosen in practice. This choice corresponds to a specific combination of weights that reflects a desired balance among the objectives. Therefore, multi-objective optimisation involves two equally important tasks: an optimisation task to find the set of Pareto optimal solutions, and a decision-making task to select the best solution from that set by choosing a specific weight combination. The latter usually requires

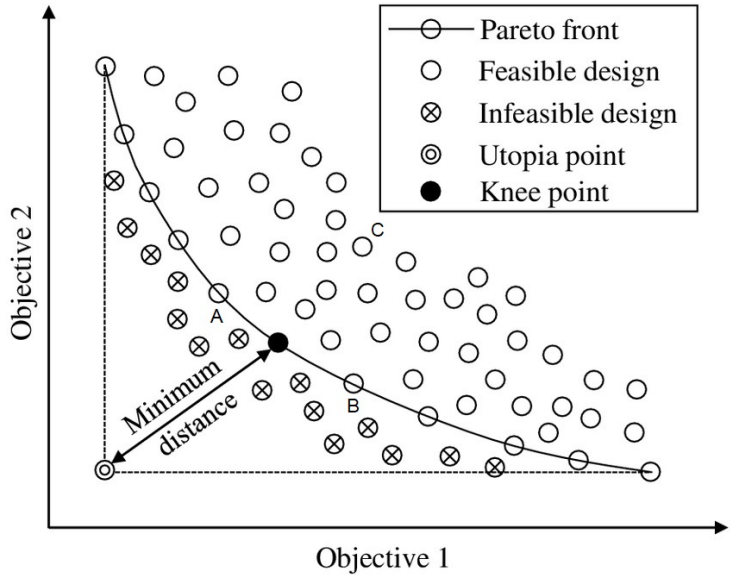

Figure 5.1: Pareto front in a two-objective minimisation problem [3]

input from a decision maker to specify their preferences and priorities. The methodology used to select this specific weight combination is described below.

5.1 Pareto front approximation

The process for discovering the Pareto front is designed to systematically explore the trade-offs between the competing objectives. Due to the complexity of the problem, it is not feasible to find the complete Pareto front. Instead, a practical approximation of the Pareto front is obtained by generating only a subset of all possible weight combinations. Furthermore, the weighted-sum method is fundamentally limited in its ability to find all Pareto optimal solutions. It is only capable of discovering supported non-dominated solutions, which lie on the convex hull of the objective space. As a result, Pareto optimal solutions that exist in non-convex regions of the objective space cannot be identified by this methodology.

Normalisation is a crucial initial step in multi-objective optimisation. This is essential for ensuring that each objective contributes to the weighted sum in a balanced way, regardless of its original units or magnitude. This allows comparison without taking their magnitude into account. To achieve this, a common reference scale must be created using two key points: the utopia point and the nadir point.

The utopia point (U) represents an ideal solution where each objective is at its absolute best possible value. This point is usually unattainable. Its value is obtained by performing

a series of individual optimisation runs, one for each objective. The results of these individual minimisations are then compiled into a payoff matrix, which shows the value of every objective for each of the single-objective optimal solutions. From this matrix, the utopia point is identified by taking the minimum value obtained for each objective across all runs. This corresponds to the diagonal of the matrix.

The nadir point (N) is a reference point that represents the worst-case objective values on the entire Pareto front. It is a point defined by the maximum value for each objective among all the Pareto optimal solutions. Since finding the complete Pareto front is computationally infeasible and the front is not fully known at this stage, an approximation of the nadir point is used instead. This approximation is found by taking the worst value for each objective from the same set of individual optimisation runs used to find the utopia point. As the objectives are being minimised, the worst value corresponds to the maximum value found for each objective.

Once these reference points are identified, all objective values are normalised to a uniform scale between 0 and 1 using the following formula:

$$obj_{norm}[I] = \frac{obj[i] - U[i]}{N[i] - U[i]}$$

Following, a large number of solutions that approximate the Pareto front are generated. This is done using the weighted-sum method, where the overall objective is to minimise a sum of the normalised objectives, each multiplied by a specific weight:

$$\min \sum_{i=1}^{3} w[i] \cdot obj_{norm}[i] \qquad \text{where} \sum_{i=1}^{3} w[i] = 1 \text{ and } w[i] \geq 0 \ \forall i \in \{1, 2, 3\}$$

By systematically varying the weight combinations, the optimisation model is solved multiple times. Each solution corresponds to a different point on the Pareto front, illustrating a unique trade-off between the objectives. This method is computationally intensive, so a limited number of weight combinations were tested. In addition, since the model takes some time to be solved, the data was limited to only three weeks. This allows the optimisation to be faster. However, it requires an extra step, which is to convert automatically the eutopia and nadir points into values for all the weeks of a term. It was estimated that they could be estimated considering that they evolve proportionally to the number of weeks. This hypothesis comes from the fact that the ratio between the initial objective for 3 rooms or for all, is approximately equal to the ratio between their number of weeks. Therefore, the nadir and utopia points are deduced from a rule of three using the number of weeks. The Pareto front for 3 weeks is represented in Figure 5.2.

All these steps are summarised in the following pseudo-code:

Algorithm 2 Pareto Front Approximation using Weighted Sum Method

Input: A multi-objective optimization model with n objectives $\{f_1, f_2, \ldots, f_n\}$
Output: An approximation of the Pareto front, $\mathcal{P}$

Initialize an empty matrix Φ of size $n \times n$.
Initialize an empty set $\mathcal{P}$ for the Pareto front.
Initialize an empty list $\mathcal{W}$ for the weights.
Generate the optimisation model.

▷ **Phase 1: Determine Utopia and Approximate Nadir Points**
for $i = 1$ to n **do**
    Set the objective to minimise f_i.
    Solve the model
    if solution is optimal **then**
        $\Phi[i] \leftarrow VALUE(objectives)$
        $\mathcal{P} \leftarrow \mathcal{P} \cup VALUE(objectives)$
        $\mathcal{W} \leftarrow$ weight vector where the i-th element is 1 and others are 0
    end if
end for
$U = \min(\Phi, \text{dim} = 2)$ ▷ Utopia point
$N = \max(\Phi, \text{dim} = 2)$ ▷ Approximate Nadir point

▷ **Phase 2: Generate Pareto Front Approximation**
Generate a set of weight combinations $\mathcal{T}$ such that $\sum_{i=1}^{n} w_i = 1$ and $w_i \geq 0$.
for each weight vector **w** in $\mathcal{T}$ **do**
    if **w** $\notin \mathcal{W}$ **then**
        Set the objective to minimize $\sum_{i=1}^{n} w_i \frac{f_i - U_i}{N_i - U_i}$.
        Solve the model
        if solution is optimal **then**
            $\Phi[i] \leftarrow VALUE(objectives)$
            $\mathcal{P} \leftarrow \mathcal{P} \cup VALUE(objectives)$
            $\mathcal{W} \leftarrow \mathcal{W} \cup w$ to
        end if
    end if
end for
Return: The set of points $\mathcal{P}$, which approximates the Pareto front.

5.2 Decision making

The final phase involves selecting the optimal set of weights. From the generated Pareto front, a decision is made to select the specific solution that represents the best balance of objective priorities. A key point which is often quite useful is the knee of the front. However, in this situation the knee worsens the travel compared to the initial allocation. Due to the nature of this research, weights which lead to a travel which increase can be discarded as interesting solutions since the main goal is to reduce this. Once that limit was imposed as illustrated in the second image of Figure 5.2, 5 weights were selected scattered across the front. For this project, a higher relative weight was assigned to minimising student travel, reflecting its primary importance. Therefore, a point with a high weight for the travel had to be chosen. The weight combination (0.2, 0.1, 0.7) was selected since it offers a good trade-off between allowing to reduce a lot the travel while trying not to worsen too much the privileged rooms and room costs.

The weights used to generate this chosen solution are then adopted for future instances, under the assumption that the relationships between objectives remain similar across similar problem instances. These weights are valid for normalised objectives, so the model objective should be adapted to take that into account. Since the normalising factors were identified when considering only 3 weeks, they must be multiplied when focusing on all weeks of the term. The multiplication is simply by the ratio of the number of weeks used since they are all proportional as illustrated in Figure 5.3.

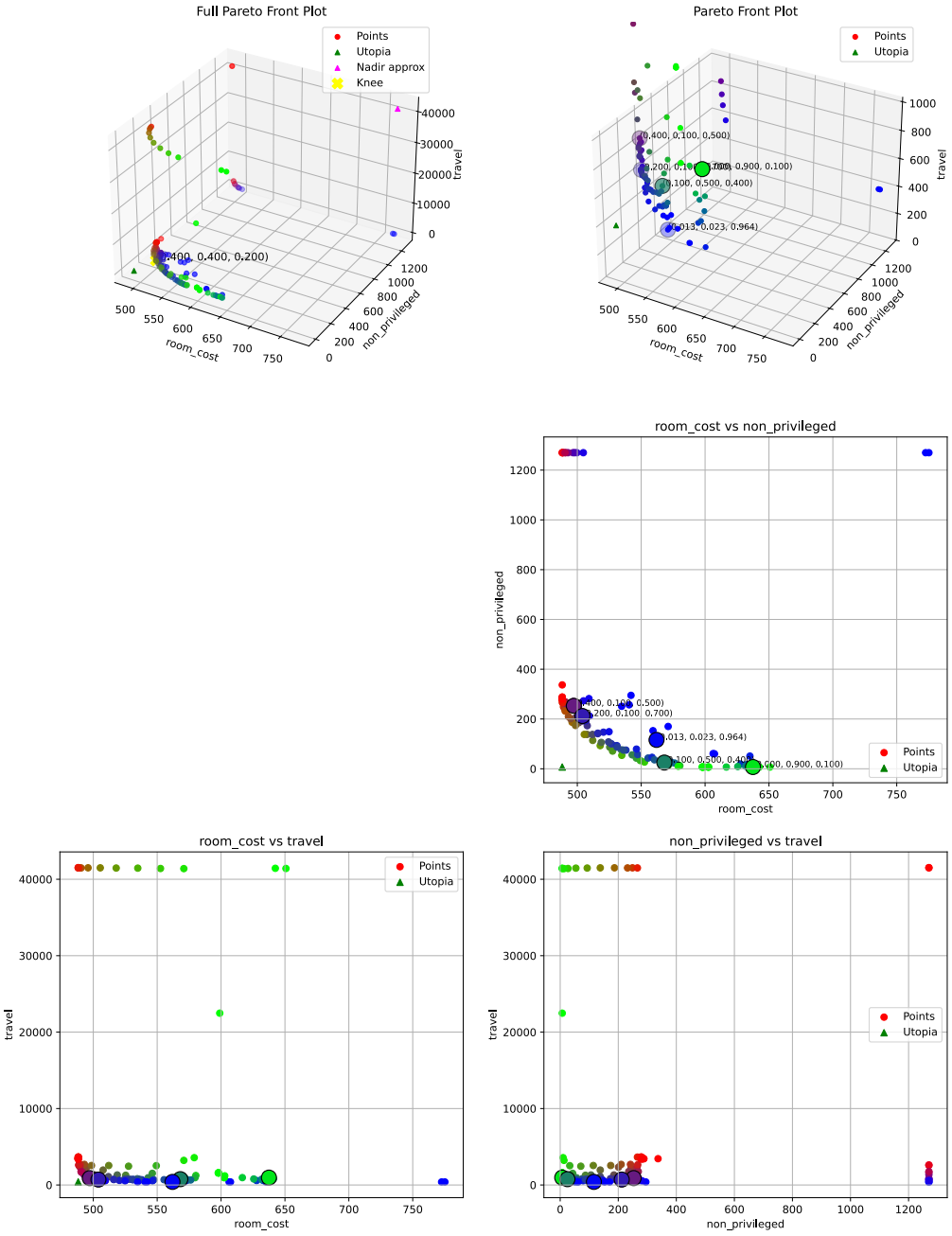

Figure 5.2: Pareto front

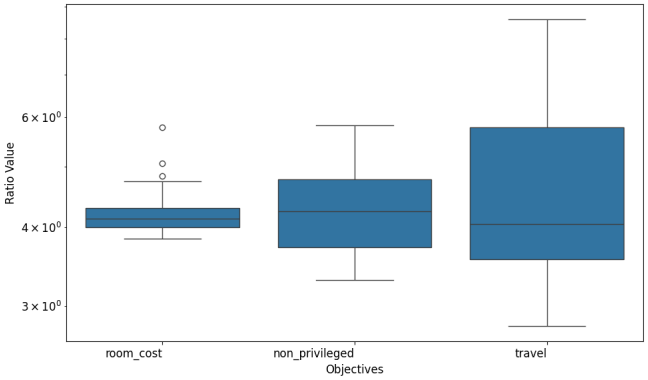

Figure 5.3: Boxplot of the ratio between 3 weeks and full term

Chapter 6

Results and discussions

6.1 Technology and hardware

The optimisation problem was implemented in Julia. The Gurobi solver version 11.0.2 was used as the MILP solver, using a free academic license. Julia's JuMP library allows to create the model and send it to the Gurobi solver. The code was executed on my personal computer equipped with an AMD Ryzen 5 4500U processor (6 cores and 6 threads) and 8 GB of RAM.

6.2 Time complexity / Objective comparison

The model was tested out in different configurations. The first one is the full model, unrestricted by the number of changes of privileged rooms, another with a limitation of 10 and another by splitting the model into blocks of 3 weeks.

Model	Optimisation time
unrestricted	1547sec
10 privileged changes	160 sec
3-week blocks	478 sec

Table 6.1: Optimisation time of the methods

It can be seen that, compared to what was expected, using 3 blocks of weeks, it a bad choice since the model takes much more time. Indeed, it had to create a lot of model which takes some times. But it can be seen that the weight have been correctly chosen

Model	room cost	non-priv	travel	total_obj_assign
unrestricted	1810	81	989	-0.001125
10 privileged changes	2097.	186	1406	0.0075304
3 weeks blocks	2097	186	1406	0.007530

Table 6.2: Objectives of the methods

and that studying 3 weeks for the pareto front then adapting it to the full term was donc correctly since they both result in the same optimal solution.

Chapter 7

Conclusions and perspectives

7.1 Conclusions

This work continues the research on course-to-room assignment for the University of Liège with the goal of reducing the travel for the students. This research introduced this problem and related problems in the literature. It then introduces basic concepts to clarify the terms used. Once all that was done, the experimentation started with the creation of a simplified version of the problem with additional constraints. With a good set of weights, this model performs in a reasonable amount of time. It was shown that it was not possible to speed up the process by splitting the model into blocks of weeks.

7.2 Perspectives

Like every research, there are limitations. These are stated in Section 3.2.3 and come from a lack of data, which introduces a lot of unrepresented constraints. If the university were to formalise these, the results could benefit from it since a faculty manager would not have to check the results.

To further reduce the travel of students, a key point that must be developed is the student group partition for each session. Digging into this partition would allow to further reduce the travel of students and could even identify courses which should be assigned to another district for these students, which would have the course before or after in the other district.

Furthermore, the approach introduced in this thesis could be extended to use when the university wants to acquire new buildings. One possibility stated in the introduction is that the University is considering changing the district of some faculty. This could allow students who follow similar courses to be together, reducing travel. By testing the program with the new rooms, it would be possible to quantify the travel that could be reduced. This could also identify which faculty would be best suited to move to these new buildings.

Bibliography

[1] Service Général d'Informatique (SEGI). https://www.segi.uliege.be/, 2025.

[2] Sara Ceschia, Roberto Maria Rosati, Andrea Schaerf, Pieter Smet, Greet Vanden Berghe, and Eugenia Zanazzo. IHTC 2024 - The Integrated Healthcare Timetabling Competition 2024. https://ihtc2024.github.io, 2024.

[3] Yang Yu, Mengfen Shen, and C. Hsein Juang. Assessing initial stiffness models for laterally loaded piles in undrained clay: Robust design perspective. *Journal of Geotechnical and Geoenvironmental Engineering*, 145(10):04019073, 2019.

[4] Découvrez les résultats de l'enquête de mobilité réalisée à l'ULiège en 2024. https://www.news.uliege.be/cms/c_20352620/fr/decouvrez-les-resultats-de-l-enquete-de-mobilite-realisee-a-l-uliege-en-2024?sc_src=email_1130253&sc_lid=119316397&sc_uid=CBaREugOKn&sc_llid=789&sc_eh=4a0574f9d030fb731, 2025.

[5] Bruno Bianchet. Enquête Cemul-ULg : Mobilité des étudiants de l'Université de Liège Principaux résultats. https://orbi.uliege.be/bitstream/2268/179459/1/Enqu%C3%AAte%20Cemul%20-%20Mobilit%C3%A9%20des%20%C3%A9tudiants%20ULg%20-%20Principaux%20r%C3%A9sultats.pdf, 2014.

[6] Wikipedia contributors. Université de liège — Wikipedia, the free encyclopedia, 2025.

[7] Le TEC. Mon nouveau réseau liégeois. https://nouveaureseau.letec.be/.

[8] United Nations. Sustainable development goal 11: Make cities inclusive, safe, resilient and sustainable. https://sdgs.un.org/goals/goal11, 2023.

[9] Aude Quinet. L'ULiège projette le retour des Facultés de Droit et Sciences humaines sur les sites d'Ethias et Chiroux: "Nous voulons réinvestir le centre-ville". https://www.lavenir.net/regions/liege/liege/2023/09/28/luliege-projette-le-retour-des-facultes-de-droit-et-sciences-humaines-au-centre-

ville-sur-les-sites-dethias-et-chiroux-4NZIDILZENDTXEQIUPEZKNTTTU/, 2023.

[10] Marc Hildesheim and Caroline Adam. L'Université de Liège envisage le retour au centre-ville des facultés de Droit et de Sciences Sociales. https://www.rtbf.be/article/l-universite-de-liege-envisage-le-retour-au-centre-ville-des-facultes-de-droit-et-de-sciences-sociales-11087151, 2022.

[11] Quentin Louveaux. A mixed-integer-programming formulation for the traveling student problem. In *ORBEL 38: 38th Annual Conference of the Belgian Operational Research Society*, Antwerp, Belgium, February 2024.

[12] Teofilo Gonzalez Sartaj Sahni. P-complete approximation problems. *Journal of the ACM*, 23, 1976.

[13] Matteo Fischetti, Michele Monaci, and Domenico Salvagnin. Three ideas for the quadratic assignment problem. *Operations Research*, 60(4):954–964, 2012.

[14] Michael W. Carter and Gilbert Laporte. Recent developments in practical course timetabling. In Edmund Burke and Michael Carter, editors, *Practice and Theory of Automated Timetabling II*, pages 3–19, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[15] Rhydian Lewis, Ben Paechter, and Olivia Rossi-Doria. *Metaheuristics for University Course Timetabling*, volume 49, pages 237–272. 01 1970.

[16] Ç Gda, Hakan Alada˘, and Gülsüm Hocao˘. A tabu search algorithm to solve a course timetabling problem. *Hacettepe Journal of Mathematics and Statistics Volume*, 36:53–64, 01 2007.

[17] E. Aycan and T. Ayav. Solving the course scheduling problem using simulated annealing. In *2009 IEEE International Advance Computing Conference*, pages 462–466, 2009.

[18] Enzhe Yu and Ki–Seok Sung. A genetic algorithm for a university weekly courses timetabling problem. *International Transactions in Operational Research*, 9(6):703–717, 2002.

[19] Nothegger, Clemens, Mayer, Alfred, Chwatal, Andreas, and Raidl, Günther R. Solving the post enrolment course timetabling problem by ant colony optimization. *Annals of Operations Research*, 194(1):325–339, 2012.

[20] Krzysztof Socha, Michael Sampels, and Max Manfrin. Ant algorithms for the university course timetabling problem with regard to the state-of-the-art. In Stefano Cagnoni, Colin G. Johnson, Juan J. Romero Cardalda, Elena Marchiori, David W. Corne, Jean-Arcady Meyer, Jens Gottlieb, Martin Middendorf, Agnès Guillot, Günther R. Raidl, and Emma Hart, editors, *Applications of Evolutionary Computing*, pages 334–345, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[21] D. de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19(2):151–162, 1985.

[22] De Werra. Construction of school timetables by flow methods. *INFOR: Information Systems and Operational Research*, 9(1):12–22, 1971.

[23] Daniel Costa. A tabu search algorithm for computing an operational timetable. *European Journal of Operational Research*, 76(1):98–110, 1994.

[24] Andrea Schaerf. Tabu search techniques for large high-school timetabling problems. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 1*, AAAI'96, page 363–368. AAAI Press, 1996.

[25] D. Abramson. Constructing school timetables using simulated annealing: Sequential and parallel algorithms. *Management Science*, 37(1):98–113, 1991.

[26] Alberto Colorni, Marco Dorigo, and Vittorio Maniezzo. Genetic algorithms: A new approach to the timetable problem. In Mustafa Akgül, Horst W. Hamacher, and Süleyman Tüfekçi, editors, *Combinatorial Optimization*, pages 235–239, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.

[27] John M. Mulvey. A classroom/time assignment model. *European Journal of Operational Research*, 9(1):64–70, 1982.

[28] R. Fahrion and G. Dollansky. Construction of university faculty timetables using logic programming techniques. *Discrete Applied Mathematics*, 35(3):221–236, 1992.

[29] Richard H. McClure and Charles E. Wells. A mathematical programming model for faculty course assignments. *Decision Sciences*, 15(3):409–420, 1984.

[30] E. M. Arkin and E. B. Silverberg. Scheduling jobs with fixed start and end times. *Discrete Appl. Math.*, 18(1):1–8, November 1987.

[31] Antoon W.J. Kolen, Jan Karel Lenstra, Christos H. Papadimitriou, and Frits C.R. Spieksma. Interval scheduling: A survey. *Naval Research Logistics (NRL)*, 54(5):530–543, 2007.

[32] Antoon W.J. Kolen and Leo G. Kroon. On the computational complexity of (maximum) class scheduling. *European Journal of Operational Research*, 54(1):23–38, 1991.

[33] R. S. MANGOUBI and DENNIS F. X. MATHAISEL. Optimizing gate assignments at airport terminals. *Transportation Science*, 19(2):173–188, 1985.

[34] Michael W. Carter and Craig A. Tovey. When Is the Classroom Assignment Problem Hard? *Operations Research*, 40(1-supplement-1):S28–S39, 1992.

[35] C. Roger Glassey and Michael Mizrach. A decision support system for assigning classes to rooms. *Interfaces*, 16(5):92–100, 1986.

[36] Karl Gosselin and Michel Truchon. Allocation of classrooms by linear programming. *Journal of the Operational Research Society*, 37(6):561–569, 1986.

[37] Gilbert Laporte and Sylvain Desroches. The problem of assigning students to course sections in a large engineering school. *Computers & Operations Research*, 13(4):387–394, 1986.

[38] Denny Kurniadi, Hendra Hidayat, Muhammad Anwar, Khairi Budayawan, Abdurrasyid Luthfi Syaifar, Zulhendra, Efrizon, and Rahmadona Safitri. Genetic Algorithms for Optimizing Grouping of Students Classmates in Engineering Education. *International Journal of Information and Education Technology*, 13(12):1907–1916, 2023.

[39] Luis Agustín-Blas, Sancho Salcedo-Sanz, Emilio Ortiz-García, Ángel Pérez-Bellido, and Antonio Portilla-Figueras. Assignment of students to preferred laboratory groups using a hybrid grouping genetic algorithm. pages 48–52, 09 2008.

[40] Kathryn A. Dowsland. *Using Simulated Annealing for Efficient Allocation of Students to Practical Classes*, pages 125–150. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.

[41] Abigail H. Chown, Christopher J. Cook, and Nigel B. Wilding. A simulated annealing approach to the student-project allocation problem. *American Journal of Physics*, 86(9):701–708, 09 2018.

[42] Ciebiera, Krzysztof and Mucha, Marcin. Student - Class Assignment Optimization Using Simulated Annealing. In *EUNIS 2014 - The 20th International Conference of European University Information Systems*, Umea, Sweden, June 2014.

[43] Kaj Holmberg. Formation of student groups with the help of optimisation. *Journal of the Operational Research Society*, 70:1–11, 01 2019.

[44] G. C. W. Sabin and G. K. Winter. The impact of automated timetabling on universities-a case study. *The Journal of the Operational Research Society*, 37(7):689–693, 1986.

[45] Decree defining the higher education landscape and the academic organization of studies. `https://gallilex.cfwb.be/sites/default/files/imports/39681_060.pdf`, 2019. French Community of Belgium, adopted on November 7, 2013, amended until May 3, 2019.

[46] Celcat. `https://www.horaires.uliege.be/`, 2025.

Appendix A

Maps of the University

Buildings from the University of Liège are split across several zones grouped into districts, which are illustrated in the following maps in Figure A.1 and Figure A.2.

Figure A.1: Map of the zones composing the district of the city centre

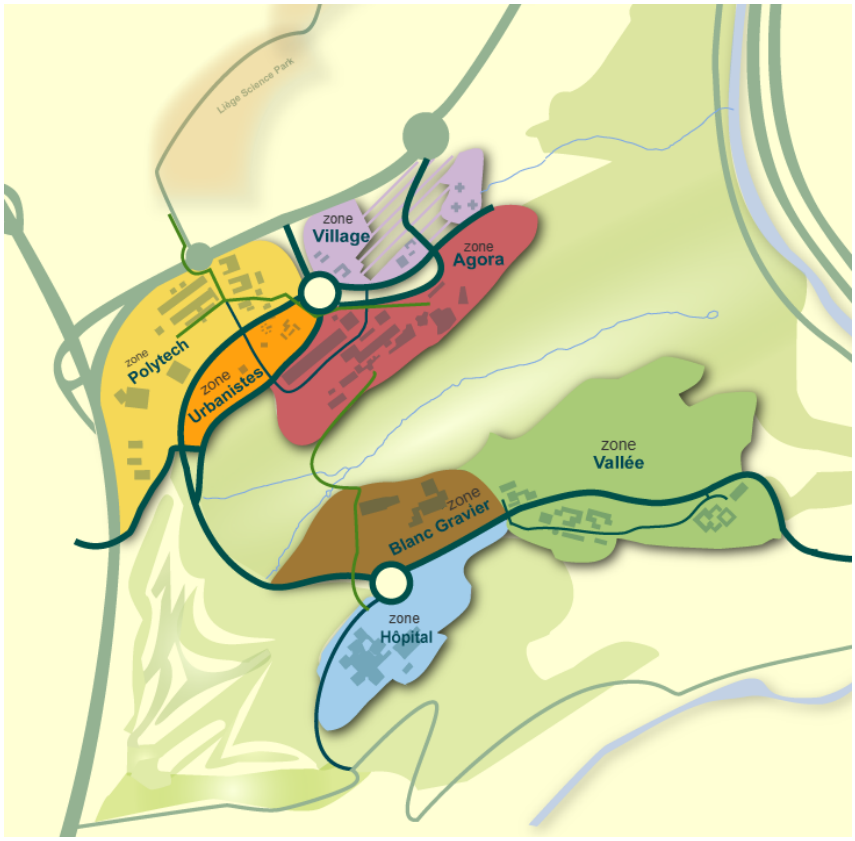

Figure A.2: Map of the zones composing the district of the Sart-Tilman

Appendix B

Files format

As mentioned in Section 4.2, the data has been combined into a single input JSON file which follows the structure of Listing B.1. The rooms allocation program also outputs a JSON file with the same format and its additional information is represented in red. The group partition program also uses that format with its annotation and modification represented in cyan. Some terms are sometimes omitted if they do not provide relevant information to have a more compact notation. For example, when "UNAVAILABLE" is empty, it was removed.

```
{
    "rooms": [
        {
            "ROOM_INDEX": int,
            "NAME": String,
            "CAPACITY": int,
            "ZONE": String,
            "DISTRICT_TRAVEL": String,
            "TYPE": String,
            "UNAVAILABLE": {
                "Q1": [
                    {
                        "DAY_OF_WEEK": int,
                        "START_TIME": String HH:MM,
                        "END_TIME": String HH:MM,
                        "WEEKS": int[]
```

```
                    }
                ],
                "Q2":  like Q1
            }
        }
    ],
    "courses": [
        {
            "COURSE_ID": int,
            "NAME": String,
            "SESSION_IDS": {
                "Q1": int[],
                "Q2": int[]
            },
            "FACULTY": String,
            "NB_STUDENTS": int,
            "STUDENTS": int[],
        }
    ]
    "sessions_Q1": [
        {
            "SESSION_ID": int,
            "COURSE_ID": int,
            "DAY_OF_WEEK": int,
            "START_TIME": String HH:MM,
            "END_TIME": String HH:MM,
            "WEEKS": int[],
            "ROOM_REQUIRED_CAPACITY": int,
            "PAST_ROOMS": int[],
            "PRIVILEGED_ROOM": int,
            "ASSIGNED_ROOMS":  int[],
            "NEW_PRIVILEGED_ROOM":  int,
            "PERCENT_STUDENTS": float,
            "NB_STUDENTS": int,
            "GROUP": int,
            "SHARED_STUDENTS": {
                "session_id int": int
            },
            "RELATED_SESSION_IDS": int[],
```

```
            "CATEGORY": true
            "ADDITIONAL_TRAVEL_PENALTY": [
                {
                    "district name": {
                        "data": [
                            {
                                "PERCENT": float,
                                "SHARED_STUDENTS_IDS": int[]
                            }
                        ],
                        "total": int
                    }
                }
            ]
        }
    ],
    "sessions_Q2": like sessions_Q1,
    "students_with_group": [
        {
            "STUDENT_ID": int,
            "GROUPS_IDS": int[][],
            "COURSES_IDS": int[][]
        }
    ],
    "groups": {
        "Q1": [
            {
                "GROUP_SET": [
                    {
                        "SESSION_ID": int,
                        "NB_STUDENTS": int,
                        "GROUP_ID": int,
                        "STUDENTS": int[],
                    }
                ],
                "INFO": {
                    "COURSE_ID": int,
                    "STUDENTS": int[]
                }
```

```
            },
        ],
        "Q2":  like Q1
    }
}
```

Listing B.1: input and output JSON format