Volume 22, Issue 2: KUS: 1361: 145-155: 02032025: 2025

KHULNA UNIVERSITY
STUDIES

journal homepage: www.kus.ku.ac.bd

Research article

Automated Class Scheduling: Ensuring Conflict-Free and Optimized Timetables

Md. Farhan Sadique, Md. Ahsanul Haque and Kazi Masudul Alam*

Computer Science and Engineering Discipline, Science Engineering and Technology School, Khulna University, Khulna – 9208, Bangladesh

ABSTRACT

Class scheduling is a crucial task for academic institutions. It requires careful allocations of classes of different courses while avoiding conflicts. Manual scheduling is often time-consuming, requires multiple revisions for optimization, and is difficult to manage when conflicts arise. In this paper, we propose an automated class scheduling system that ensures conflict-free schedules while maximizing gaps between consecutive classes of the same course, where possible. Our system takes into account key scheduling constraints, such as class duration, the number of classes per course per week, the gap between the classes of each course, conflicts from overlapping classes within the same semester or for the same course teacher, and the avoidance of scheduling classes during breaks or after the designated end time of the academic day. We validate our approach using real-world academic data from a discipline at Khulna University, demonstrating its practicality and efficiency. The results show that our system effectively eliminates scheduling conflicts and reduces administrative workload. Our solution is ready for immediate implementation in any academic institution.

ARTICLE INFO

Article timeline:
Date of Submission:
02 March, 2025
Date of Acceptance:
28 December, 2025
Article available online:
29 December, 2025

***Keywords*:**
Automated Class Scheduling
Class Routine Generator
Conflict-Free Scheduling
Timetable Optimization
Scheduling Algorithm
Academic Resource Management

Introduction

At the beginning of each semester/term, or academic year, educational institutions schedule their classes. This process considers various factors, including course offerings, the number of classes per course, class duration, and course teachers. A valid schedule must avoid conflicts such as multiple classes of the same course or batch overlapping, or a teacher being assigned to multiple classes at the same time. Given these constraints, manually generating a class schedule is complex and time-consuming. An automated class scheduling system addresses this challenge by efficiently generating schedules while meeting all required criteria.

The study of automated class scheduling began in 1968 (Stewart & Clark, 1968), to the best of our knowledge. Since then, several approaches have been explored. An integer linear programming-based university class scheduling system was proposed by Ahmed Wasfy and Fadi A. Aloul (Wasfy & Aloul, 2007). Genetic algorithms are widely used to address this problem (Li & Aickelin, 2008; Rao, Sravani, Kumar, Kishore, & Sri Venkata Ramana, 2024; Guevara Sánchez, Sánchez Rojas, & Aguilar-Alonso, 2024; Ji & Qiu, 2023; Ikhwani, Marzuki, & Ramadhan, 2022; Roy, Kabir, & Ahmed, 2022; Chen, Yue, Li, Zhumadillayeva, & Liu, 2021; Kakkar, et al., 2021). In recent years, machine learning has been widely applied across various domains, including class scheduling. Ermis Soumalias et al. (Soumalias, Zamanlooy, Weissteiner, & Seuken, 2022) proposed a machine learning-based course allocation system.

Although many studies have been conducted on class scheduling, they often fail to account for all possible conflict scenarios. Even when some conflicts are considered, most approaches are not entirely conflict-free. In practice, several types of conflicts may arise during class scheduling, including overlaps among classes of the same academic term, overlaps between classes assigned to the same teacher, conflicts with break times or the end of the day, and uneven distribution of classes within a day or across the week. In this work, we propose a class scheduler that systematically considers all possible conflict scenarios in a general educational institution. The proposed scheduler also optimizes the placement of classes for each course throughout the week to ensure an even distribution of the classes of a course across days.

The key contributions of our class scheduling system are:

*Corresponding author: < kazi@cseku.ac.bd >
DOI: https://doi.org/10.53808/KUS.2025.22.02.1361-se

- Classes for each course are spaced out as much as possible throughout the week.
- No idle hours occur between classes of different courses unless there is a conflict.
- Extra holidays are scheduled if the number of required classes is insufficient.
- No conflicts arise from scheduling multiple classes for students of the same academic term/semester at the same time. This also ensures that no two classes are scheduled in the same room simultaneously.
- No conflicts occur due to assigning multiple classes of the same teacher at the same time.
- Real-world validation using academic data from Khulna University.
- A practical solution that can be used in various institutions.

The rest of this paper is organized as follows: the related works are discussed, the proposed methodology is detailed, experimental results are presented, and the study is concluded.

Literature Review

Ch. Venkateswara Rao et al. (Rao, Sravani, Kumar, Kishore, & Sri Venkata Ramana, 2024) proposed a class scheduling system using reinforcement learning and genetic algorithms. This system utilizes previous class schedules along with current data to generate an optimized routine. It prioritizes faculty satisfaction. It requires 45 hours to complete scheduling, resulting in five scheduling conflicts. These conflicts include issues such as overlapping classes due to a teacher being assigned multiple classes at the same time and multiple classes being scheduled in the same classroom; however, these issues are rare in this system. It achieves 85% room utilization.

Several genetic algorithm-based class scheduling systems exist. Georgette Alexia Guevara Sánchez et al. (Guevara Sánchez, Sánchez Rojas, & Aguilar-Alonso, 2024) discussed some of these systems and developed a web application for genetic algorithm-based class scheduling. They do not guarantee a conflict-free class routine, and they did not report any results regarding room utilization.

Dawei Ji and Yang Qiu (Ji & Qiu, 2023) applied a combination of the Genetic Algorithm (Zhang, et al., 2021) and the Firefly Algorithm (Yang, 2009) for class scheduling, referring to their approach as the improved Firefly Algorithm. Although it is a faster approach, it lacks the ability to generate fully conflict-free routine.

An automated lecture scheduling system using an Evolutionary Algorithm (EA) is proposed by Yusri Ikhwani et al. (Ikhwani, Marzuki, & Ramadhan, 2022). This system distributes classes in a balanced way, ensuring there are no idle hours between them. It strictly follows hard constraints, such as students fitting in the classroom, a classroom being used for only one lecture at the same time, and lectures of the same course being scheduled at different times, but soft constraints related to teaching-learning activities between lectures may occasionally be broken. Therefore, it is not entirely conflict-free.

During any pandemic (e.g., the COVID-19 pandemic), conducting classes from home becomes a common practice. However, this requires each student to have a device (computer or mobile) to attend classes. In households with multiple students, this requirement increases. Eman Mukhaimer (Mukhaimer, 2022) proposed a class scheduling system using integer programming that takes into account the number of available devices and the number of students in a household.

Genetic algorithm-based web application that helps create class schedules automatically was proposed by Sujit Roy et al. (Roy, Kabir, & Ahmed, 2022). The system allows management teams to enter schedules for different student batches. It then automatically generates timetables for teachers, classrooms, and labs.

The study by Xiangliu Chen et al. (Chen, Yue, Li, Zhumadillayeva, & Liu, 2021) tested the mutation genetic algorithm for course scheduling. This work eradicates the shortcomings of hill climbing algorithms, tabu search algorithms, ant colony algorithms, and simulated annealing algorithms for class scheduling.

Mohit Kumar Kakkar et al. (Kakkar, et al., 2021) proposed Genetic Algorithm (GA) and Memetic Algorithm (MA) for generating class scheduling. They introduced a smart adaptive mutation scheme to improve the process. This work compares GA and MA to see which one works better in reducing scheduling conflicts. The results show that GA is faster and more adaptable.

Quantum Evolutionary Algorithms (QEA) based timetabling problem was proposed by Mohammad-H. Tayarani-N. (Tayarani-N, 2021). It introduces two new operators to improve QEA. They are reinitialization operator to reset the population when it gets stuck and diversity-preserving operator to help individuals explore different areas.

Tarek Sobh et al. (Sobh, Cousens, & Patel, 2007) proposed an automated course scheduler which considers student needs, instructor availability, and institutional rules to create the best schedules. It uses the SKED program (Mihali, Sobh, & Vamoser, 2004) to check students' progress and decide which courses they need to graduate quickly.

Methodology

Several scheduling parameters are maintained in our method. These include classroom allocation (by keeping one classroom for each academic term), academic term, teacher assignment, class duration, class days, break times, class start times, the number of classes per course, distribution of classes evenly across the week, and minimization of idle hours between classes to create additional free days for students. Below, we describe the approach for maintaining these parameters as part of our method.

Our class scheduling method requires a list of all courses, eligible class times and days, and break times. Each course includes the following information:

- The term/semester to which the course belongs
- Course identification number, referred to as the course no.
- Course type (Theory/Sessional). This information is useful for displaying in the generated class routine. However, it is not required for class scheduling because the duration of sessional course classes is generally longer than that of theory course classes, and the number of sessional classes per week is

usually fewer than that of theory classes. Therefore, the number of classes per week is used as a factor to schedule fewer sessional classes and more theory classes within a week. To allocate a greater number of slots for sessional classes and a smaller number of slots for theory classes, the duration of each class is used.

- Duration of each class in hours. Generally, the duration of each class depends on the credit value of the course. Courses with higher credit values have longer class durations. For example, a sessional course with a credit value of 1 may have a class duration of 1.5 hours, whereas a sessional course with a credit value of 1.5 may have a class duration of 2.5 hours. Hence, class duration is not fixed by course type (theory or sessional) and can be set individually for each course by the user.
- Number of classes per week.
- Teacher's name.

Class times refer to the list of eligible start times for classes. Generally, classes start on the hour or at half-hour intervals. The eligible start times are: 09:00, 09:30, 10:00, 10:30, 11:00, 11:30, 12:00, 12:30, 13:00, 13:30, 14:00, 14:30, 15:00, 15:30, 16:00, and 16:30, assuming the day starts at 09:00 and ends at 17:00. These values are flexible and can be modified by the user.

Class days are all weekdays except the weekly holidays. The class days are Sunday, Monday, Tuesday, Wednesday, and Thursday, while Friday and Saturday are designated as weekly holidays. These days can be adjusted according to the user's preference.

A slot is a specific time period during which a class is held. Slots begin at the specified class start times. The class routine consists of all the slots where courses are scheduled without conflicts.

Scheduling with Even Distribution

For each term, we start placing the classes of the courses into available slots one by one. The classes of a course are evenly distributed across the week. For example, if a course has three classes per week, it would not be feasible for students to attend all three classes on the same day. The greater the gap between classes of a course, the more time students have to grasp the previous lesson and prepare for the next one. Therefore, in this case, each class is scheduled with a two-day gap (e.g., Sunday, Tuesday, and Thursday), provided that slots are available on those days without conflicts. We define the gap (in number of days) between each class of a course as shown in Table 1.

For each course, the first class is scheduled in the earliest available slot of the week. The first slot is at 09:00 on Sunday, which is the first class start time on the first day of the week. The second class (if applicable) is scheduled in the first available slot after the designated gap (as specified in Table 1), and this process continues for subsequent classes. If no slot is available on a given day, the first available slot on the next day is used. If the next day also lacks an available slot, the scheduling continues to the following day, and so on, until a suitable slot is found.

Table 1: Gap Between Classes Based on the Number of Classes per Week

Number of Classes per Week	**Gap**
1	Not applicable
2	4 days
3	2 days
4 or 5	1 day

Since we are scheduling the classes of one term first, followed by the classes of another term, the classes of each term will either be grouped together (classes of different courses) or placed in separate blocks (because of the gap between the classes of a course) in the class routine. This results in free time slots and may even provide extra holidays for students if the total number of classes is insufficient.

Availability of Slots

A slot is available for a course if no conflicts exist for that course. Conflicts can arise in the following scenarios:

- Conflict 1: Multiple classes within the same term are scheduled in overlapping slots. We assume that each term has a separate classroom, so only one class per term can take place at a time.
- Conflict 2: A teacher is scheduled to teach multiple classes at overlapping times.
- Conflict 3: A class is scheduled during break time. Break start time and break time duration are provided by the user.
- Conflict 4: A class ends after the last scheduled time of the day, which is 17:00 (5:00 PM), depending on the user input.

The end of a class (i) is defined using Equation 1.

$$end_i = start_i + Duration\ of\ class\ in\ hours_i \quad (1)$$

Conflicts 1 can occur in the following cases (illustrated in Figure 1):

- A class (i) ends inside another class (j) of the same term (Slot 1 of Figure 1, Equation 2).

$$(start_j \leq end_i \leq end_j) \wedge (start_i < start_j) \quad (2)$$

- A class (i) starts inside another class (j) of the same term (Slot 2 of Figure 1, Equation 3).

$$(start_j \leq start_i \leq end_j) \wedge (end_j < end_i) \quad (3)$$

- A class (i) starts and ends inside another class (j) of the same term (Slot 3 of Figure 1, Equation 4).

$$(start_j \leq start_i \leq end_j) \wedge (start_j \leq end_i \leq end_j) \quad (4)$$

- A class (i) starts before another class (j) of the same term and ends after it (Slot 4 of Figure 1, Equation 5).

$$(start_i < start_j) \wedge (end_j < end_i) \quad (5)$$

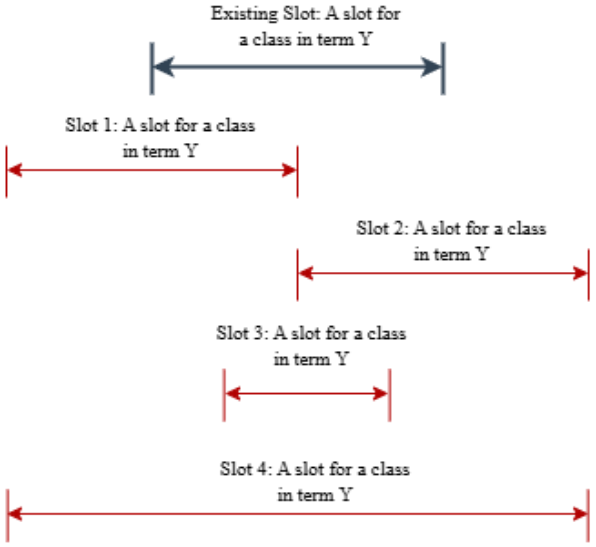

Figure 1: Conflict Scenarios Due to Overlapping Slots Within the Same Term

We examine the validity of Equation 6 to determine whether conflict 1 exists between classes i and j.

$$(end_i < start_j) \vee (end_j < start_i) \quad \forall i, j \; such \; that \; term/classroom_i = term/classroom_j, day_i = day_j, i \neq j \quad (6)$$

Conflicts 2 can occur in the following cases (illustrated in Figure 2):

- A class (***i***) ends inside another class (***j***) of the same teacher, regardless of the term (Slot 1 of Figure 2, Equation 2).
- A class (***i***) starts inside another class (***j***) of the same teacher, regardless of the term (Slot 2 of Figure 2, Equation 3).
- A class (***i***) starts and ends inside another class (***j***) of the same teacher, regardless of the term (Slot 3 of Figure 2, Equation 4).
- A class (***i***) starts before another class (***j***) of the same teacher and ends after it, regardless of the term (Slot 4 of Figure 2, Equation 5).

We examine the validity of Equation 7 to determine whether conflict 2 exists between classes ***i*** and ***j***.

$$(end_i < start_j) \vee (end_j < start_i) \quad \forall i, j \; such \; that \; teacher_i = teacher_j, day_i = day_j, i \neq j \quad (7)$$

The end of break time is defined using Equation 8.

$$break_{end} = break_{start} + Duration \; of \; break \quad (8)$$

Conflicts 3 can occur in the following cases (illustrated in Figure 3):

- A class (***i***) ends inside break time (Slot 1 of Figure 3, Equation 9).

$$(break_{start} \leq end_i \leq break_{end}) \wedge (start_i < break_{start}) \quad (9)$$

- A class (i) starts inside break time (Slot 2 of Figure 3, Equation 10).

$$(break_{start} \leq start_i \leq break_{end}) \wedge (break_{end} < end_i) \quad (10)$$

- A class (i) starts and ends inside break time (Slot 3 of Figure 3, Equation 11).

$$(break_{start} \leq start_i \leq break_{end}) \wedge (break_{start} \leq end_i \leq break_{end}) \quad (11)$$

- A class (i) starts before break time and ends after it (Slot 4 of Figure 3, Equation 12).

$$(start_i < break_{start}) \wedge (break_{end} < end_i) \quad (12)$$

We examine the validity of Equation 13 to determine whether conflict 3 exists for a class i.

$$(end_i < break_{start}) \vee (break_{end} < start_i) \quad \forall i \quad (13)$$

Conflict 4 can occur when a class (i) starts before the last scheduled time of the day and ends after it (slot 1 of Figure 4, Equation 14).

$$start_i < day_{end} < end_i \quad (14)$$

We examine the validity of Equation 15 to determine whether conflict 4 exists for a class i.

$$(end_i \leq day_{end}) \quad \forall i \quad (15)$$

Figure 5 shows the flowchart illustrating our class scheduling procedure. The implementation of our class scheduling approach is available at https://github.com/iammfsr/automated-class-scheduler.

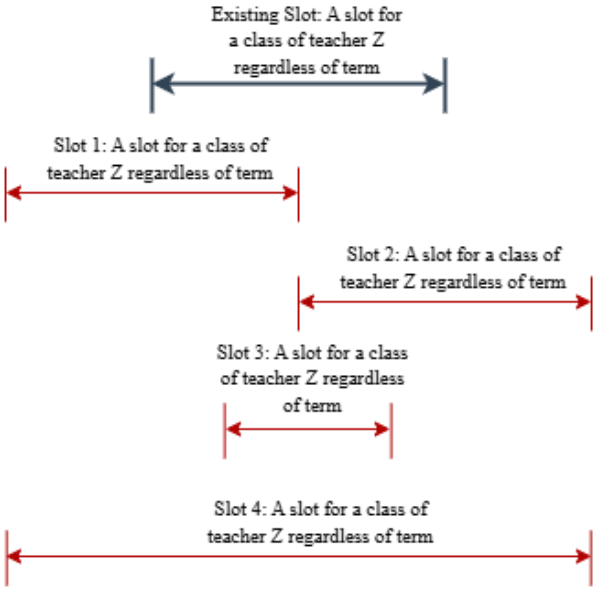

Figure 2: Conflict Scenarios Due to Overlapping Slots of a Teacher's Classes

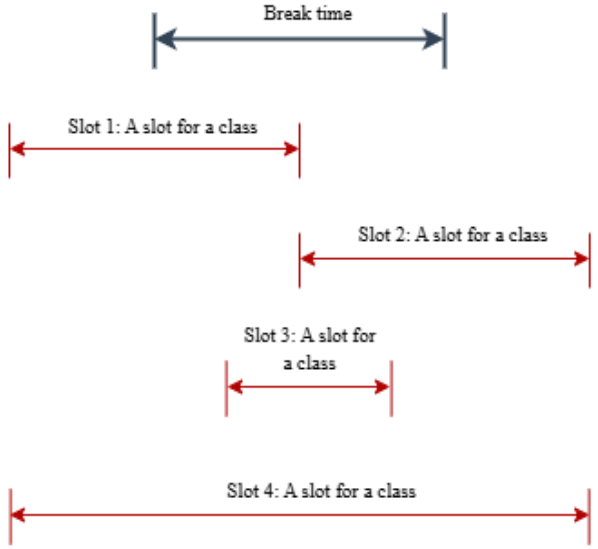

Figure 3: Conflict Scenarios Caused by Overlapping Slots with Break Time

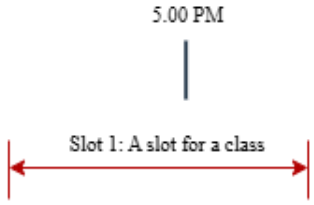

Figure 4: Conflict Scenarios Due to Overlapping Slots with the Final Scheduled Time of the Day

Conflict Decision Parameters and Their Effects

Class start time and class duration are the primary parameters that cause time overlap. Increasing class duration increases the likelihood of Conflict 1 (term or classroom-based conflict) and Conflict 2 (teacher-based conflict) due to the reduced number of available time slots. Conflicts 3 (break time-based conflict) and 4 (end time-based conflict) are also more likely to occur, as longer class durations increase the probability of classes extending into break periods or beyond the designated end time. When a teacher is assigned many classes, conflicts are more likely to occur among that teacher's classes (Conflict 2). Increasing break time also increases the likelihood of conflicts with class times (Conflict 3). If the end of the day is set too early, Conflict 4 becomes more common. An increased number of classes per course affects all conflict types by reducing scheduling flexibility. An increased number of concurrently running academic terms can increase Conflict 2. However, it does not increase Conflict 1, as we generally assume that each running academic term has a separate classroom.

Results Analysis

Our class scheduling system arranges all classes within the smallest number of days possible in a week. This optimization provides additional holidays for students when there are not enough classes to fill the entire week. The system achieves this by placing classes of different courses in the earliest available slots, ensuring that consecutive days are fully utilized while keeping as many free days as possible. Additionally, it minimizes idle hours between classes unless necessary to avoid conflicts (any of the conflicts 1, 2, 3, and 4 discussed in the *Availability of Slots* subsection).

To maintain a balanced schedule, our system distributes classes of a course at the maximum possible interval within the week by strategically spacing them out (Table 1). This helps ensure an even distribution of a course's classes, allowing students sufficient breathing space between sessions of the same course.

By addressing Conflict 1 (as discussed in the *Availability of Slots* subsection), our system prevents exceeding the number of available classrooms. Moreover, it ensures that no scheduling conflicts occur due to multiple classes of the same academic term or the same course being scheduled at the same time.

Similarly, by resolving Conflict 2 (as discussed in the *Availability of Slots* subsection), our system ensures that teachers' classes are scheduled in conflict-free time slots.

Finally, resolving Conflicts 3 and 4 (as discussed in the *Availability of Slots* subsection) guarantees that no classes are scheduled during break times or beyond the designated end time of the day.

Generated Schedule for a Dataset

We collected data on the ongoing undergraduate courses of the Computer Science and Engineering Discipline at Khulna University, Khulna, during the second half of 2024. These courses span multiple academic terms, including 1st year 1st term (1-1), 2nd year 1st term (2-1), 2nd year 2nd term (2-2), 3rd year 2nd term (3-2), and 4th year 2nd term (4-2). The details of each course in the dataset follow the format shown in Table 2. The class start times follow the format shown in Table 3, and the class days follow the format shown in Table 4.

The generated class schedule using our system, which considers a break time from 13:00 to 14:00, is presented in Tables 5, 6, 7, 8, and 9.

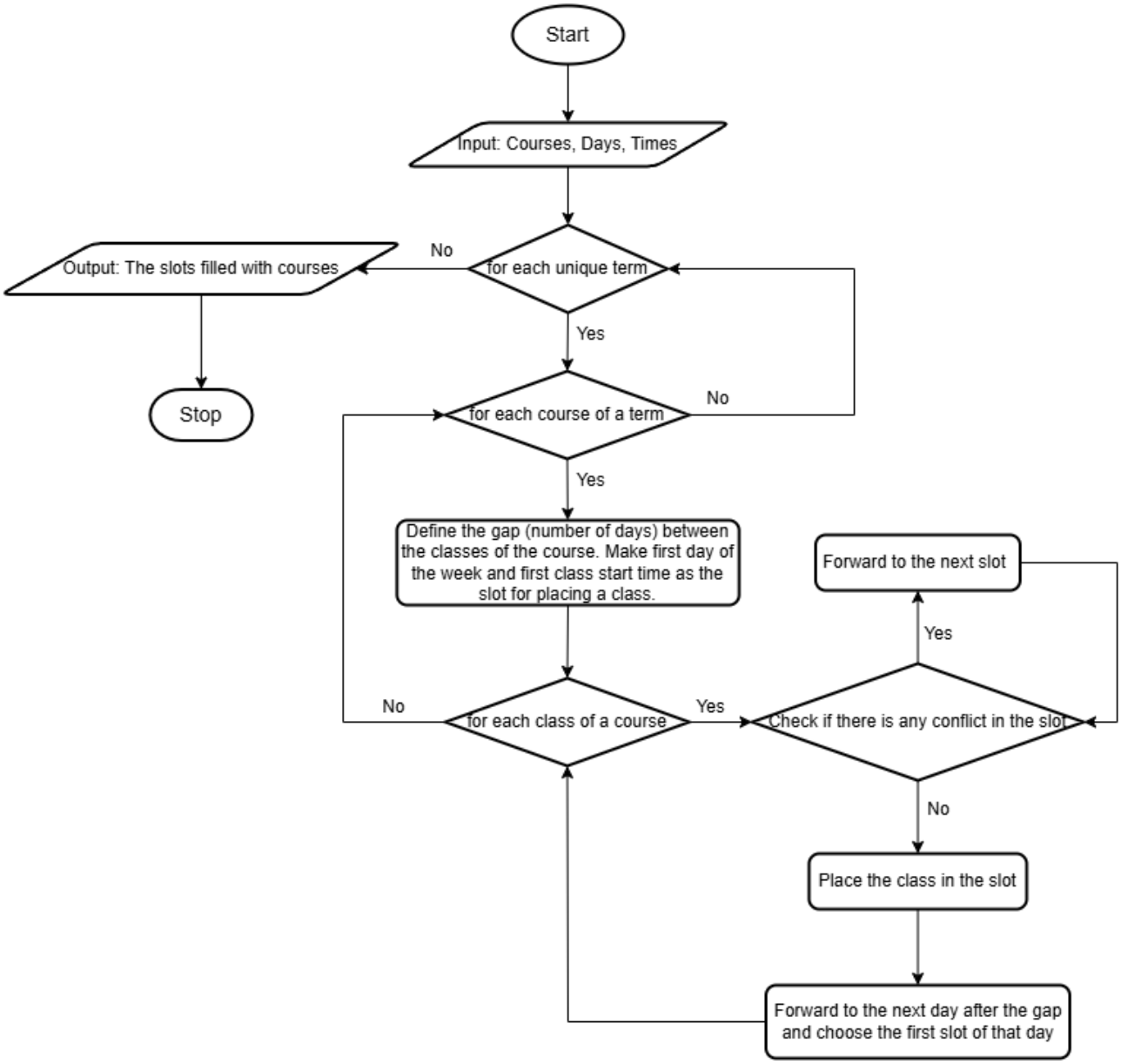

Figure 5: Flowchart of Our Class Scheduling Process

Table 2: Format of Course Details in the Dataset

Term	Course No.	Course Type	Duration of each class in hours	Number of classes per week	Teacher's name

Dataset of the courses:

```
1-1 071402CSE1100 Sessional 1.30 1 SC
1-1 071402CSE1101 Theory 1.00 3 MM
1-1 071402CSE1102 Sessional 2.15 1 MM
1-1 071402CSE1103 Theory 1.00 3 ASMA
1-1 054102Math1151 Theory 1.00 3 SFA
1-1 053302Phy1151 Theory 1.00 3 SB
1-1 053302Phy1152 Sessional 1.30 1 SB
1-1 023102Eng1151 Theory 1.00 2 X
1-1 023102Eng1152 Sessional 1.30 1 X
2-1 071402CSE2100 Sessional 2.15 1 KMA
2-1 071402CSE2101 Theory 1.00 3 MFS
2-1 071402CSE2102 Sessional 2.15 1 MFS
```

```
2-1 071402CSE2103 Theory 1.00 3 SC
2-1 071402CSE2104 Sessional 1.30 1 SC
2-1 071402CSE2105 Theory 1.00 3 MAR
2-1 071402CSE2106 Sessional 2.15 1 MAR
2-1 071402ECE2151 Theory 1.00 3 MMH
2-1 071402ECE2152 Sessional 1.30 1 MMH
2-1 054102Math2151 Theory 1.00 3 UKM
2-2 071402CSE2201 Theory 1.00 3 AKB
2-2 071402CSE2202 Sessional 2.15 1 AKB
2-2 071402CSE2203 Theory 1.00 3 AS
2-2 071402CSE2204 Sessional 1.30 1 AS
2-2 071402CSE2205 Theory 1.00 3 AI
2-2 071402CSE2206 Sessional 2.15 1 AI
2-2 071402ECE2251 Theory 1.00 2 Y
2-2 054102Math2251 Theory 1.00 3 LEA
3-2 071402CSE3201 Theory 1.00 3 AKM
3-2 071402CSE3202 Sessional 1.30 1 AKM
3-2 071402CSE3204 Sessional 1.30 1 SAH
3-2 071402ECE3251 Theory 1.00 3 MSA
3-2 071402ECE3252 Sessional 1.30 1 MSA
3-2 041302BA3251 Theory 1.00 2 Z
3-2 042102Law3251 Theory 1.00 2 Q
3-2 031302Psy3251 Theory 1.00 2 P
3-2 071402CSE3225 Theory 1.00 3 RD
3-2 071402CSE3226 Sessional 1.30 1 RD
4-2 071402CSE4202 Sessional 2.15 1 MM
4-2 071402CSE4204 Sessional 1.30 1 ASMA
4-2 041302BA4251 Theory 1.00 2 FA
4-2 071402CSE4231 Theory 1.00 3 KHT
4-2 071402CSE4232 Sessional 1.30 1 KHT
4-2 071402CSE4243 Theory 1.00 3 GMAR
4-2 071402CSE4223 Theory 1.00 3 AKM
4-2 071402CSE4224 Sessional 1.30 1 AKM
```

Table 3: Format of Class Start Times in the Dataset

Id	Class start time

Dataset of the class start times:

```
10 09:00
20 09:30
30 10:00
40 10:30
50 11:00
60 11:30
70 12:00
80 12:30
90 13:00
100 13:30
110 14:00
120 14:30
130 15:00
140 15:30
150 16:00
160 16:30
```

Table 4: Format of Class Days in the Dataset

Id	Class day

Dataset of the class days:
1000 Sunday
2000 Monday
3000 Tuesday
4000 Wednesday
5000 Thursday

Table 5: Class Schedule on Sunday (Generated by Our System)

Day	Term	Course No.	Teacher's Name	Start time – End time
Sunday	1-1	071402CSE1100	SC	09:00 - 10:30
		071402CSE1101	MM	10:30 - 11:30
		071402CSE1102	MM	14:00 - 16:15
		071402CSE1103	ASMA	11:30 - 12:30
	2-1	071402CSE2100	KMA	09:00 - 11:15
		071402CSE2101	MFS	11:30 - 12:30
		071402CSE2102	MFS	14:00 - 16:15
	2-2	071402CSE2201	AKB	09:00 - 10:00
		071402CSE2202	AKB	10:00 - 12:15
		071402CSE2203	AS	14:00 - 15:00
		071402CSE2204	AS	15:00 - 16:30
	3-2	071402CSE3201	AKM	09:00 - 10:00
		071402CSE3202	AKM	10:00 - 11:30
		071402CSE3204	SAH	11:30 - 13:00
		071402ECE3251	MSA	14:00 - 15:00
		071402ECE3252	MSA	15:00 - 16:30
	4-2	071402CSE4204	ASMA	09:00 - 10:30
		041302BA4251	FA	10:30 - 11:30
		071402CSE4231	KHT	11:30 - 12:30
		071402CSE4232	KHT	14:00 - 15:30
		071402CSE4243	GMAR	15:30 - 16:30

Table 6: Class Schedule on Monday (Generated by Our System)

Day	Term	Course No.	Teacher's Name	Start time – End time
Monday	1-1	054102Math1151	SFA	09:00 - 10:00
		054102Math1151	SFA	10:00 - 11:00
		053302Phy1151	SB	11:00 - 12:00
		053302Phy1151	SB	12:00 - 13:00
		053302Phy1152	SB	14:00 - 15:30
		023102Eng1151	X	15:30 - 16:30
	2-1	071402CSE2103	SC	09:00 - 10:00
		071402CSE2103	SC	10:00 - 11:00
		071402CSE2104	SC	11:00 - 12:30
		071402CSE2105	MAR	14:00 - 15:00
		071402CSE2105	MAR	15:00 - 16:00
		071402ECE2151	MMH	16:00 - 17:00
	2-2	071402CSE2205	AI	09:00 - 10:00
		071402CSE2205	AI	10:00 - 11:00
		071402CSE2206	AI	14:00 - 16:15
		071402ECE2251	Y	11:00 - 12:00
		071402ECE2251	Y	12:00 - 13:00
	3-2	041302BA3251	Z	09:00 - 10:00
		041302BA3251	Z	10:00 - 11:00
		042102Law3251	Q	11:00 - 12:00
		042102Law3251	Q	12:00 - 13:00
		031302Psy3251	P	14:00 - 15:00
		031302Psy3251	P	15:00 - 16:00
		071402CSE3225	RD	16:00 - 17:00
	4-2	071402CSE4202	MM	09:00 - 11:15
		071402CSE4223	AKM	11:30 - 12:30
		071402CSE4223	AKM	14:00 - 15:00
		071402CSE4224	AKM	15:00 - 16:30

Table 7: Class Schedule on Tuesday (Generated by Our System)

Day	Term	Course No.	Teacher's Name	Start time – End time
Tuesday	1-1	071402CSE1101	MM	09:00 - 10:00
		071402CSE1103	ASMA	10:00 - 11:00
		023102Eng1151	X	11:00 - 12:00
		023102Eng1152	X	14:00 - 15:30
	2-1	071402CSE2101	MFS	09:00 - 10:00
		071402CSE2106	MAR	10:00 - 12:15
		071402ECE2151	MMH	14:00 - 15:00
		071402ECE2152	MMH	15:00 - 16:30
	2-2	071402CSE2201	AKB	09:00 - 10:00
		071402CSE2203	AS	10:00 - 11:00
		054102Math2251	LEA	11:00 - 12:00
		054102Math2251	LEA	12:00 - 13:00
	3-2	071402CSE3201	AKM	09:00 - 10:00
		071402ECE3251	MSA	10:00 - 11:00
		071402CSE3225	RD	11:00 - 12:00
		071402CSE3226	RD	14:00 - 15:30
	4-2	071402CSE4231	KHT	09:00 - 10:00
		071402CSE4243	GMAR	10:00 - 11:00

Table 8: Class Schedule on Wednesday (Generated by Our System)

Day	Term	Course No.	Teacher's Name	Start time – End time
Wednesday	1-1	054102Math1151	SFA	09:00 - 10:00
		053302Phy1151	SB	10:00 - 11:00
	2-1	071402CSE2103	SC	09:00 - 10:00
		071402CSE2105	MAR	10:00 - 11:00
		071402ECE2151	MMH	11:00 - 12:00
		054102Math2151	UKM	12:00 - 13:00
		054102Math2151	UKM	14:00 - 15:00
		054102Math2151	UKM	15:00 - 16:00
	2-2	071402CSE2205	AI	09:00 - 10:00
	3-2	071402CSE3225	RD	09:00 - 10:00
	4-2	071402CSE4223	AKM	09:00 - 10:00

Table 9: Class Schedule on Thursday (Generated by Our System)

Day	Term	Course No.	Teacher's Name	Start time – End time
Thursday	1-1	071402CSE1101	MM	09:00 - 10:00
		071402CSE1103	ASMA	10:00 - 11:00
	2-1	071402CSE2101	MFS	09:00 - 10:00
	2-2	071402CSE2201	AKB	09:00 - 10:00
		071402CSE2203	AS	10:00 - 11:00
		054102Math2251	LEA	11:00 - 12:00
	3-2	071402CSE3201	AKM	09:00 - 10:00
		071402ECE3251	MSA	10:00 - 11:00
	4-2	041302BA4251	FA	09:00 - 10:00
		071402CSE4231	KHT	10:00 - 11:00
		071402CSE4243	GMAR	11:00 - 12:00

From Tables 5, 6, 7, 8, and 9, we can see that the class schedule generated by our system distributes the classes of the courses across the week as evenly as possible (as discussed in the *Scheduling with Even Distribution* subsection) while ensuring a completely conflict-free schedule (as discussed in the *Availability of Slots* subsection).

Comparison with Existing Works

Table 10 compares our class scheduling system with some recent related works.

Table 10: Comparative Analysis of Related Works and Our Approach

Criteria	**OptiTime** (Rao, Sravani, Kumar, Kishore, & Sri Venkata Ramana, 2024)	**Web Application Based on Genetic Algorithms** (Guevara Sánchez, Sánchez Rojas, & Aguilar-Alonso, 2024)	**Course Scheduling Based on Improved FA Algorithm** (Ji & Qiu, 2023)	**Lecture Scheduling Based on Evolutionary Algorithm** (Ikhwani, Marzuki, & Ramadhan, 2022)	**Our Class Scheduling System**
Balanced class distribution of a course	Mostly	Mostly	Mostly	Mostly	Yes
Classes are placed on the minimum number of days / No unnecessary idle hours between classes	Yes	Did not consider	Mostly	Yes	Yes
Conflict because of placing multiple classes of the same course	Did not consider	No	Did not consider	Did not consider	No
Conflict because of placing multiple classes of the same term / More number of classes at a time than the number of class rooms	Rare	Rare	Rare	Rare	No
Conflict because of placing multiple classes of the same teacher at the same time	No	No	No	No	No
Conflict because of placing a class on break times or beyond the designated end time of the day	Rare	Did not consider	Did not consider	Did not consider	No

Our class scheduling system meets all the criteria mentioned in Table 10, whereas other recent works do not satisfy all the criteria.

Conclusion

Our automated class scheduling system maximizes gaps between classes of a course while ensuring a completely conflict-free schedule, making it suitable for real-world applications. The generated schedule, tested on a real course dataset from a discipline at Khulna University, demonstrates its practical applicability. However, incorporating teacher preferences for convenient time slots could further enhance its effectiveness. This study presents a simple yet highly practical scheduling system that can be readily implemented in any institution.

Conflict of Interest

The authors declare that they have no conflicts of interest.

References

Chen, X., Yue, X.-G., Li, R. Y., Zhumadillayeva, A., & Liu, R. (2021). Design and Application of an Improved Genetic Algorithm to a Class Scheduling System. *International Journal of Emerging Technologies in Learning (iJET), 16*(1), 44-59. Retrieved from https://www.learntechlib.org/p/218642/

Guevara Sánchez, G. A., Sánchez Rojas, J. A., & Aguilar-Alonso, I. (2024). Implementation of a Web Application Based on Genetic Algorithms for the Preparation of Academic Schedules in an Educational Unit. *Nanotechnology Perceptions, 20*(4), 792–804. Retrieved from https://nano-ntp.com/index.php/nano/article/view/813

Ikhwani, Y., Marzuki, K., & Ramadhan, A. (2022). Automated University Lecture Schedule Generator based on Evolutionary Algorithm. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer, 22*(1), 129-138. Retrieved from https://www.researchgate.net/publication/379016563_Automated_University_Lecture_Schedule_Generator_based_on_Evolutionary_Algorithm

Ji, D., & Qiu, Y. (2023). Course Scheduling in Vocational Education Colleges Based on Improved FA Algorithm Under Multi-Objective Constraints. *2023 7th Asian Conference on Artificial Intelligence Technology (ACAIT)* (pp. 410-415). Jiaxing: IEEE. Retrieved from https://ieeexplore.ieee.org/abstract/document/10528463

Kakkar, M. K., Singla, J., Garg, N., Gupta, G., Srivastava, P., & Kumar, A. (2021). Class Schedule Generation using Evolutionary Algorithms. *Journal of Physics: Conference Series, 1950*. Retrieved from https://iopscience.iop.org/article/10.1088/1742-6596/1950/1/012067/meta

Li, J., & Aickelin, U. (2008). Explicit Learning: An Effort towards Human Scheduling Algorithms. *arXiv preprint arXiv:0804.0580*. Retrieved from https://arxiv.org/abs/0804.0580

Mihali, R., Sobh, T., & Vamoser, D. (2004). SKED: A course scheduling and advising software. *Computer Applications in Engineering Education, 12*(1), 1-19. doi:https://doi.org/10.1002/cae.10054

Mukhaimer, E. (2022). *Pandemic-Based School Classes Scheduling Automation Algorithm: Palestinian Schools as a Testing Case*. Master's Thesis, An-Najah National University, Nablus. Retrieved from https://repository.najah.edu/items/6b4dfa25-b295-4574-8ea9-84eea856bc4e

Rao, C., Sravani, K. V., Kumar, L. S., Kishore, M. N., & Sri Venkata Ramana, M. D. (2024). OptiTime: AI-Powered Faculty Scheduler for Peak Productivity. *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)* (pp. 1-6). Chennai: IEEE. Retrieved from https://ieeexplore.ieee.org/abstract/document/10533576

Roy, S., Kabir, M. H., & Ahmed, M. T. (2022). Design and Implementation of Web-based Smart Class Routine Management System for Educational Institutes. *International Journal of Education and Management Engineering (IJEME), 12*(2), 38-48. Retrieved from https://www.mecs-press.org/ijeme/ijeme-v12-n2/v12n2-5.html

Sobh, T., Cousens, R., & Patel, S. (2007). *Course Scheduler: An Automated Schedule Generator*. University of Bridgeport, Department of Computer Science and Engineering, Connecticut. Retrieved from https://www.researchgate.net/publication/256456000_Course_Scheduler_An_Automated_Schedule_Generator

Soumalias, E., Zamanlooy, B., Weissteiner, J., & Seuken, S. (2022). Machine Learning-Powered Course Allocation. *arXiv preprint arXiv:2210.00954*. Retrieved from https://arxiv.org/abs/2210.00954

Stewart, J., & Clark, R. L. (1968). University of Maryland Student Scheduling Algorithm. *1968 23rd ACM National Conference* (pp. 555–562). New York: Association for Computing Machinery. Retrieved from https://dl.acm.org/doi/abs/10.1145/800186.810619

Tayarani-N, M.-H. (2021). Novel Operators for Quantum Evolutionary Algorithm in Solving Timetabling Problem. *Evolutionary Intelligence, 14*, 1869-1893. Retrieved from https://link.springer.com/article/10.1007/s12065-020-00438-0

Wasfy, A., & Aloul, F. A. (2007). Solving the University Class Scheduling Problem Using Advanced ILP Techniques. *IEEE GCC Conference*, (pp. 1-5). Retrieved from https://www.researchgate.net/profile/Fadi-Aloul/publication/228634502_Solving_the_University_Class_Scheduling_Problem_Using_Advanced_ILP_Techniques/links/02e7e5259692e8d824000000/Solving-the-University-Class-Scheduling-Problem-Using-Advanced-ILP-Technique

Yang, X.-S. (2009). Firefly Algorithms for Multimodal Optimization. *Stochastic Algorithms: Foundations and Applications* (pp. 169-178). Springer, Berlin, Heidelberg. doi:https://doi.org/10.1007/978-3-642-04944-6_14

Zhang, H., Thompson, J., Gu, M., Jiang, X. D., Cai, H., Liu, P. Y., . . . and Liu, A. Q. (2021). Efficient On-Chip Training of Optical Neural Networks Using Genetic Algorithm. *ACS Photonics, 8*(6), 1662-1672. doi:https://doi.org/10.1021/acsphotonics.1c00035