# Evolutionary Allocation with Dynamic Guidance for Pre-Scheduled Timetables

Anniza Hamdan[1], Sze San Nah[2], Goh Say Leng[3], Emily Sing Kiang Siew[4]
Faculty of Computer Sciences and Mathematics,
MARA University of Technology, Kota Samarahan, Sarawak 94300, Malaysia[1]
Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak,
Kota Samarahan, Sarawak 94300, Malaysia[2]
Optimization and Visual Analytics Research Group-Faculty of Computing and Informatics,
Universiti Malaysia Sabah, Labuan International Campus, Labuan 87000, Malaysia[3]
i-CATS University College, Kuching, Sarawak 93350, Malaysia[4]

***Abstract*—The University Course Timetabling Problem (UCTP) is a well-known combinatorial optimization challenge that involves allocating courses to timeslots and rooms while satisfying various institutional constraints. At other institutions, general courses (e.g., language subjects) are prioritised during timetable allocation due to their high enrolment numbers from multiple faculties. However, at the MARA University of Technology, Sarawak Branch, Mathematics and Statistics (MAT/STA) courses are shared across several programs, with timeslot availability limited by pre-scheduled major courses in each program. This study presents a tailored evolutionary algorithm for a real institutional scenario, incorporating dynamic local search and guided variation operators to improve feasibility and solution quality. A case study was conducted using real datasets from the Department of Mathematical Sciences at MARA University of Technology, Sarawak Branch. Benchmark datasets from ITC2002 and ITC2007 (Track 2) are employed for comparative evaluation against existing methods. The algorithm successfully produced a feasible timetable and outperformed manually prepared schedules in terms of soft-constraint penalties. Results indicate strong performance on real datasets, producing a high-quality, balanced timetable with reduced preparation time, particularly in handling repeating students, lecturers' availability, and limited timeslots. Nevertheless, lower performance on benchmark datasets suggests the need for hybridization or additional operators to handle larger, more complex problems. Overall, the results demonstrate the effectiveness of the proposed approach in real-world timetabling while maintaining acceptable quality on benchmark instances.**

***Keywords*—Timetabling problem; evolutionary algorithm; dynamic; guided**

## I. Introduction

The UCTP is a well-known NP-hard optimization challenge that involves assigning courses, rooms, timeslots, and student groups under various hard and soft constraints [1], [2], [3]. The constraints of UCTP vary across academic institutions due to their respective policies and regulations. Practically, the goal of UCTP is to provide a balanced timetable with minimal conflicts. Hard conflicts may cause an infeasible timetable, such as when a lecturer or students have two courses scheduled at the same time. Soft conflicts determine the quality of the timetable, such as a morning or afternoon class overload, and too many consecutive lecture hours in a day.

This research is motivated by a different practice of course timetabling in MARA University of Technology (UiTM), Sarawak Branch. The Department of Mathematical Sciences at UiTM Sarawak Branch offers MAT/STA courses across 14 programs, including Public Administration, Accountancy, Engineering, and more. Current timetabling prioritizes the allocation of major courses (of programs). Each major course may have multiple weekly sessions, for example, 2 hours on Day 2, 2 hours on Day 3, and 1 hour on Day 4 (2+2+1). The pattern of these multiple weekly sessions can be 3+0, 2+2, or 3+1. Consequently, the available slots of 2+2 MAT/STA courses are limited. In addition, each lecturer must satisfy minimum teaching hours and shared teaching requirements. At the same time, parallel timeslots for MAT/STA courses are necessary to accommodate students who repeat across multiple program groups.

Timetable preparation begins after the final examination of each semester. The projected numbers of repeat students and enrolments are used to estimate the required groups for each MAT/STA course. A program-level group projection (may take 1-2 weeks) must be completed first. Once receiving these projections, the Department of Mathematical Sciences may further adjust the number of required groups for each MAT/STA course based on teaching load and shared-teaching requirements for the semester. The existing timetabling system allows users to view available slots and manually assign MAT/STA courses to program or group timetables. Nonetheless, since multiple programs take these courses, timetable conflicts must be checked manually. Some lecturers may request fewer consecutive teaching hours in a day, while others may express concerns about the balance between morning and afternoon teaching slots. Repeat students are also facing issues with their repeated courses. These adjustments and checks continue until the semester starts (commonly stable after week 4 of lectures).

Furthermore, the allocation of the MAT/STA course can adversely affect timetable quality by creating an imbalance in daily load. For instance, a pre-scheduled timetable already contains six lecture hours on days 1, 2, and 3. Assigning additional sessions to these days increases the consecutive hours per day. Redistributing the MAT/STA course to the lighter days 4 and 5 would promote better load balancing and enhance overall timetable efficiency. Once the semester begins, further adjustments may compromise the stability and feasibility of the established timetable. Therefore, an efficient strategy is required to address this issue.

Motivated by real-life applications, the problem aims to minimize the conflicts while providing a balanced timetable in a short time. Three semesters of data are collected. An evolutionary algorithm (EA) is proposed to explore possibilities for MAT/STA course allocations while minimizing conflicts and achieving a balanced timetable. There are previous studies that have been done for real-life applications using EA variants: Genetic algorithm [4], [5]; Particle Swarm Optimization [1], [2], [3], [6]; and Hybrid Evolutionary Algorithm [7], [8], [9]. Most of these studies introduce new operators (crossover and mutation) during the evolutionary stage to address the nature of real-world applications. Some operators allow temporary conflicts during evolution and refine them later. Hybridization with local search was performed to refine the solution. Hence, the proposed algorithm incorporates a dynamic local operator and guided variation operators to refine the search process and improve timetable quality adaptively.

## II. Formulation and Approach

A timetable is represented as a three-dimensional (3D) array (groups/lecturer, day, and timeslot). A binary decision variable, $X_{c,t,r}$, is used (which takes the value 1) when a lecture of the course $c$ is allocated to a day $d$ and timeslot $t$. Given that $R_{t,r} = 1$ indicates that a room is counted for each timeslot $t$. All MAT/STA courses for the assigned groups must be scheduled each week into available timeslots and rooms. Multiple weekly sessions of 2+2 MAT/STA lecture hours are fully allocated, and each lecturer teaches only one MAT/STA course per timeslot and room. Sufficient rooms are available for all multiple weekly sessions, and no MAT/STA courses are assigned to blocked timeslots. Blocked timeslots are defined as follows:

- Last timeslot of each day, $t = \{9,18,27,36,45\}$.
- Lunch hour, 1 p.m. until 2 p.m. (not included in a set of timeslots).
- Extra co-curricular activity, Wednesday 4 p.m. until 6 p.m. for year 1 students (not included in a set of timeslots).
- Friday prayer, Friday 12 noon until 2 p.m. (not included in a set of timeslots).

To achieve a balanced, high-quality timetable, MAT/STA courses with a 2+2 lecture hours should be evenly distributed throughout the week, with minimal preset gaps. The students' group should not have more than 7 consecutive lecture hours per day, and isolated 2-hour MAT/STA lectures should be avoided. Penalties are applied whenever these preferences are violated.

To the best of our knowledge, there is no real-world application of previous studies using an evolutionary algorithm that considers a session gap in course allocation with remaining timeslots (major courses of a program are scheduled first). Maintaining the crucial session gap while allocating courses to limited time slots complicates the standard UCTP. The most relevant paper [9] used an improved genetic algorithm on the benchmark dataset ITC2007 (Track 2). However, the allocation of events in ITC2007 (Track 2) is based solely on available timeslots, without any specific session gaps between events.

An EA is proposed in this study due to its flexibility in handling diverse constraints of the problem. It allows interaction and information exchange in a population-based environment [10]. The main steps of EA consist of the following: 1) Initialization, 2) Fitness evaluation, 3) Parents selection for reproduction, 4) Variation operators, 5) Selection of the best solution [11].

As the case study includes an additional hard constraint of a 2+2 session gap for each MAT/STA course, a dynamic local operator is specifically designed and applied during initialization and variation operators. Excluding this operator will produce infeasible timetables. A steady-state algorithm generates a population size of [20,100] at the initial stage. The improvement stage evolves this population iteratively using variation operators, generating a complete, updated solution.

### *A. Initialization*

In EA, random initialization may lead to an infeasible solution when constraints are not enforced. Rezaeipanah et al. [9] applied a random shuffling technique to promote genetic diversity and population diversity. Nevertheless, in this study, we apply sorting and shuffling techniques to ensure that the high-constraint student groups are prioritized.

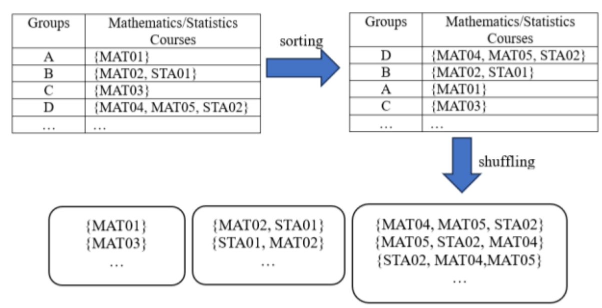


Fig. 1. Sorting and shuffling technique in the initialization phase.

Fig. 1 illustrates the MAT/STA courses ranked by the number of courses taken by each group. The MAT/STA courses in this sorted list are shuffled to produce different initial allocations. The iteration continues until the desired population size is achieved.

If any conflicts are found, a dynamic local operator is used to adjust the preset session gap for MAT/STA courses; it is necessary to obtain a feasible initial solution. The adjustment is made by reducing the preset session gap value for the course, which may result in a session gap penalty later on. Nonetheless, due to the highly constrained MAT/STA groups, some runs may not achieve the maximum population size.

### *B. Fitness Evaluation*

Let $X_i, i = 1,2,3, \dots P$, denote a candidate timetable in a population P. The solution quality is measured based on a weighted soft-constraint penalty objective function:

$$F_{obj}(X) = w_1 \sum_{c \in C} Q_c + w_2 \sum_{g \in G} \sum_{d=1}^{5} C_{g,d} + w_3 \sum_{g \in G} \sum_{d=1}^{5} I_{g,d}. \quad (1)$$

In Eq. (1), $Q_c$ measures session-gap violations for the course $c$. $C_{g,d}$ denotes excess consecutive hours for the group $g$ on day $d$, and $I_{g,d}$ counts any isolated sessions. The coefficients $w_1, w_2, w_3$ are fixed values of penalty weights for each soft constraint violation. Each soft-constraint violation type employs a predefined penalty weight with a predefined cost: $w_1 = 1$ for session-gap violations, $w_2 = 7$ for excessive consecutive hours, and $w_3 = 1$ for an isolated MAT/STA course within a day. The objective is to minimize $F_{obj}(X)$, where smaller values indicate higher timetable quality.

One advantage of a sum-weighted penalty function is that it can be applied to any optimization technique, as it focuses on minimizing a single objective function. In addition, it is flexible and easy to use, as the weight can be assigned based on its importance [12]. In the existing literature [13], [14], [15], the cost penalty for 3 consecutive hours is set to 1. Meanwhile, we significantly increased the value of $w_2$ to highlight the severity of pre-scheduled timetables. For the real datasets, the counting of consecutive hours also accounted for pre-scheduled slots on the same day. Consequently, allocating the MAT/STA course to the days that already contain long consecutive hours could negatively affect the overall quality and balance of the timetable.

## C. Parents Selection

A $k$-tournament selection is used at this stage with $k = 3$. Three individual timetables are selected randomly and compared (fitness value). The timetable with the lowest penalty value is chosen as Parent 1. The process is repeated to select Parent 2, provided that Parent 2 is not the same individual as Parent 1. Fig. 2 illustrates the selection process for the Parent.

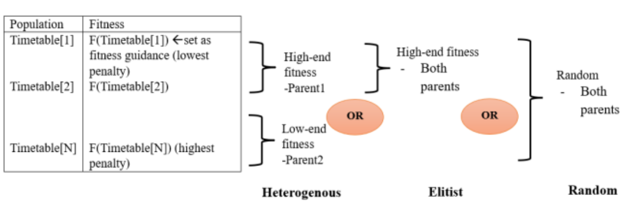


Fig. 2. Illustration of parent selection mechanisms used in the evolutionary process.

The proposed algorithm employs three types of selection strategies: 1) Heterogeneous, 2) Elitist, and 3) Random. Individuals in the population are ranked before selection. In Heterogeneous selection, Parent 1 is randomly selected from the high-end fitness individuals, and the other is randomly selected from the low-end fitness individuals to preserve diversity and balance exploration and exploitation. In Elitist selection, both parents are randomly selected from the high-end fitness individuals. Meanwhile, in Random selection, parents are selected randomly from the population.

## D. Swapping 1 (External)

Previous studies have applied a single-point, uniform heuristic crossover and other variation techniques, such as swap sequence, forceful swap, switching tolerance, horizontal, and vertical swap, to generate offspring. Similarly, Swapping 1 purposely exchanges two candidate solutions (Parent 1: $x_i$ and Parent 2: $x_p$) by matching a random event with the same MAT/STA course. This guided approach reduces randomness and complexity caused by limited timeslots, while same-course swapping helps avoid conflicts.

In candidate $x_i$, Session 1 of course $c_i$, scheduled on day 2 in timeslots 3-4, is swapped with Session 1 of the same course code in the timetable $x_p$, scheduled on day 1 in timeslots 1-2. However, Session 2 cannot be swapped directly because it conflicts with major courses. A dynamic local operator is activated to handle this conflict. Two offspring ($x'_i$ and $x'_p$) are obtained. Fig. 3 demonstrates the external swapping between Parent 1 and Parent 2.

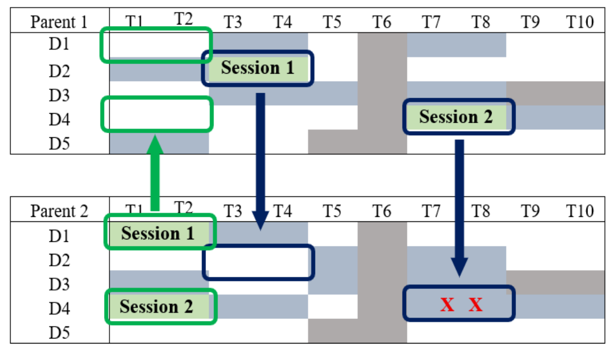


Fig. 3. External swap of two parent solutions, where selected sessions are exchanged. the dynamic local operator handles the resulting conflict ("XX").

## E. Swapping 2 (Internal)

Swapping 2 functions like a mutation operator by performing a guided internal swap in each offspring ($x'_i, x'_p$) generated from Swapping 1. For each offspring, a random 2-hour session with the same MAT/STA course is selected. Session 1 in $x'_{i1}$ (day 1, timeslots 3-4) is swapped with Session 1 in $x'_{i2}$ (day 2, timeslots 9-10). As the timeslots in $x'_{i2}$ are unavailable, the dynamic local operator reallocates the swapped Session 2 to the next available timeslots. Penalties are recalculated, and the offspring replaces the parent if it improves the solution. Fig. 4 illustrates the internal swapping in each offspring.

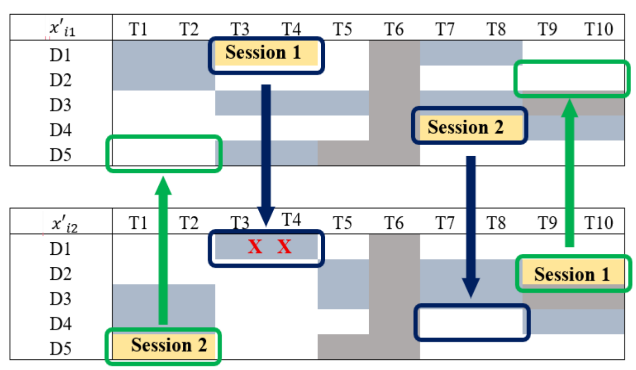


Fig. 4. Internal session swap within each offspring solution, where selected sessions are exchanged, and the resulting conflict ("xx") is repaired using the dynamic local operator.

## F. Checking Termination Criteria

The algorithm is terminated when a maximum number of generations is reached, or no improvement is observed over a predefined number of consecutive generations.

### G. Selection of the Best Solution

After the termination criteria are satisfied, the candidate solutions are ranked by their penalty values (objective function costs). The solution with the lowest penalty value is chosen as the final solution. The average penalty value is calculated from the solutions of the last generation.

### H. Guided Variation Operator

In other institutions, general courses such as language subjects, which are taken by all programs, are prioritized during timetable allocation due to their complexity. In contrast, at the UiTM Sarawak Branch, a MAT/STA course is offered across multiple programs, with each program comprising several groups enrolled in the same MAT/STA course. Given this scenario, the same MAT/STA course was chosen for swapping to minimize the complexity of finding feasible timeslots, as variations in pre-scheduled major courses across groups increase timetabling complexity.

TABLE I. NUMBER OF GROUPS IN EACH PROGRAM TAKING THE MAT/STA COURSE.

| Course Code | Program Code | Student Year | Number of groups | | |
|---|---|---|---|---|---|
| | | | October 2022 | March 2023 | October 2023 |
| **Diploma Programs** | | | | | |
| MAT111 | BA132 | 1 | 5 | 4 | 7 |
| MAT112 | AC110 | 1 | 6 | 1 | 5 |
| | AM110 | 1 | 7 | 5 | 7 |
| | IC110 | 1 | 4 | 2 | 4 |
| MAT133 | AS125 | 1 | 1 | 1 | 1 |
| | AS126 | 1 | 2 | 1 | 2 |
| | AS120 | 1 | 6 | 2 | 4 |
| | CS110 | 1 | 7 | 5 | 7 |
| MAT183 | EH110 | 1 | 4 | 0 | 2 |
| | EC110 | 1 | 3 | 1 | 3 |
| | EE111 | 1 | 0 | 0 | 1 |
| | AS125 | 2 | 0 | 1 | 0 |
| | AS120 | 2 | 0 | 7 | 1 |
| | CS110 | 2 | 1 | 8 | 4 |
| MAT210 | CS110 | 3 | 6 | 3 | 8 |
| MAT235 | EC110 | 1 | 0 | 2 | 1 |
| | EE111 | 1 | 0 | 1 | 0 |
| | EH110 | 1 | 0 | 2 | 1 |
| MAT238 | AS120 | 3 | 4 | 1 | 1 |
| MAT285 | EC110 | 3 | 4 | 1 | 1 |
| STA104 | AC110 | 2 | 1 | 9 | 1 |
| STA108 | AS115/ AS120 | 3 | 2 | 6 | 2 |
| STA116 | CS110 | 3 | 2 | 6 | 2 |
| **Degree Programs** | | | | | |
| MAT402 | AM228 | 1 | 2 | 1 | 2 |
| MAT421 | AS222 | 1 | 1 | 0 | 1 |
| STA404 | AM228 | 2 | 5 | 6 | 5 |
| | AS222 | 3 | 1 | 0 | 0 |
| | AC220 | 3 | 1 | 2 | 1 |
| STA408 | AT220 | 2 | 1 | 3 | 1 |
| | AS222 | 1 | 0 | 0 | 1 |
| ITS665 | AS222 | 3 | 1 | 0 | 1 |

Table I indicates the number of groups taking one or more MAT/STA courses. The difficulty of identifying feasible timeslots increases as more programs enrol in the MAT/STA course or when a program takes multiple MAT/STA courses (see Fig. 5).

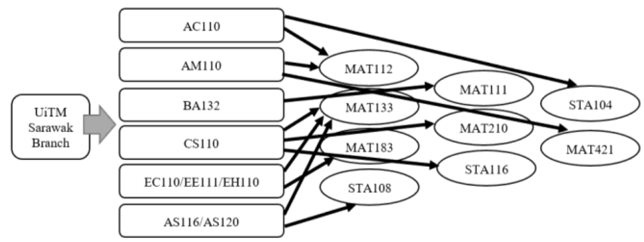


Fig. 5. Complexity of MAT/STA courses taken by multiple programs.

### I. Dynamic Local Operator

This operator is crucial in Initialization, Swapping 1, and Swapping 2. While variation operators generate offspring to explore the solution space, this operator acts as a dynamic solver that resolves conflicts arising from guided variations, particularly in handling the session gaps. In the Initialization stage, a preset session gap is used to schedule a second session of a 2-hour MAT/STA course after the first one is successfully allocated. However, pre-scheduled major courses may occupy certain time slots, creating conflicts. Thus, Session 2 is relocated to the next available timeslot, and the session gap is recalculated. To prevent sessions from being scheduled too closely together, a penalty is applied when the updated gap is smaller than the preset session gap. Fig. 6 illustrates the relocation process for Session 2 (with a preset session gap of 18 timeslots).

The same process is applied in Swapping 1 and Swapping 2. Nevertheless, conflicts encountered during these swapping operations may necessitate relocating either or both of Sessions 1 and 2. This adaptive strategy ensures feasibility while permitting controlled relaxation to enhance exploration and minimise unallocated courses.

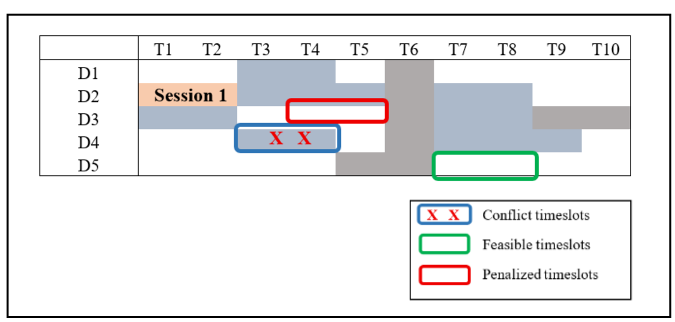


Fig. 6. Relocation process in the initialization stage.

## III. Experiments

### A. Experimental Setting

In this case study, a real-world dataset is derived from an actual university timetable from the UiTM, Sarawak Branch. A real challenge and diverse policies across organizations in this dataset are studied to tailor the proposed algorithm and evaluate its performance. The details of the data are summarized in the following Table II:

TABLE II. CHARACTERISTICS OF REAL DATASETS FOR THREE SEMESTERS.

| Semester | October 2022 | March 2023 | October 2023 |
|---|---|---|---|
| No. of programs | 15 | 15 | 15 |
| No. of students | 1637 | 1372 | 1408 |
| No. of groups | 78 | 67 | 81 |
| No. of MAT/STA courses | 14 | 14 | 16 |
| No. of groups taking multiple MAT/STA courses | 1(2 courses) | 11(2 courses) | 5(2 courses) 1(3 courses) |
| No. of lecturers | 21 | 23 | 23 |
| No.of MAT/STA courses with shared-teaching | 0 | 7 | 3 |

The experiment was coded in Java, and all experiments were carried out on an Acer computer equipped with a 12th Gen Intel® Core™ i5-1235U processor running at 3.45 GHz. We performed 10 independent runs for each dataset. In addition, the ITC2002 and ITC2007 (Track 2) benchmark datasets were used as a baseline to evaluate the proposed algorithm's performance relative to existing methods.

In preliminary experiments, various settings were used to determine the best session gaps and end timetable hours for each dataset. The details of the setting tested are as follows:

- Session gaps: {9, 10, 11, 12, 13, 14, 15, 16, 17, 18}
- End timetable: {17, 18, 19, 20}

Preliminary experiments show that the best predefined session gaps for each semester are 11, 13, and 13, respectively. Nonetheless, the session gap for each dataset may vary for multiple MAT/STA courses due to adjustments made by the dynamic local operator.

### B. Experimental Results

Table III indicates that penalty values (average of soft-constraint violations) were reduced by 56.19% in October 2022, 66.30% in March 2023, and 59.63% in October 2023, showing substantial improvements in lecture distribution, consecutive hours, and isolated sessions. Although some datasets required later end times and more timeslots, this trade-off enabled a more balanced distribution of parallel sessions and lower penalties in a more complex environment. Overall, the results demonstrate the method's effectiveness in generating a feasible and balanced timetable while addressing critical session gaps.

The algorithm performance was further evaluated in terms of the penalty cost achieved in the initial and improvement stages (see Table IV). Random initial solutions may take time to improve during an evolutionary stage [16]. A sorting and shuffling technique was applied in the initial stage which indicates the effectiveness of obtaining good-quality feasible solutions. The proposed guided variation operator also speeds up the improvement of each solution. The significant improvement is evident from the large deviation in penalty cost reduction during the improvement stage.

TABLE III. OVERALL PERFORMANCE (AVERAGE COST) OF THREE DATASETS.

| Semester | October, 2022 | | March, 2023 | | October, 2023 | |
|---|---|---|---|---|---|---|
| | **Act.** | **Imp.** | **Act.** | **Imp.** | **Act.** | **Imp.** |
| End timetable | 18:00 | 18:00 | 17:00 | 18:00 | 17:00 | 19:00 |
| Penalty Value | 105 | 46 **↓56.19%** | 181 | 61 **↓66.30%** | 218 | 88 **↓59.63%** |
| Timeslots | 46 | 46 | 42 | 46 | 42 | 50 |

Act. = Actual; Imp. = Improved

TABLE IV. STATISTICAL ANALYSIS FOR THE INITIAL STAGE AND IMPROVEMENT STAGE ACROSS THREE DATASETS.

| Datasets | Stage | Minimum | Mean | Std. Deviation |
|---|---|---|---|---|
| **October, 2022** | Initial | 21 | 52.16 | 10.88 |
| | Improvement | 0 | 46 | **20.93** |
| **March, 2023** | Initial | 49 | 68.54 | 8.17 |
| | Improvement | 0 | 61 | **21.86** |
| **October, 2023** | Initial | 98 | 143.32 | 13.71 |
| | Improvement | 7 | 88 | **33.58** |

A one-way analysis of variance (ANOVA) based on relative improvement was conducted, as shown in Table V, to evaluate the algorithm's stability across independent runs. The results indicate consistent algorithm performance across all datasets and multiple runs (p-value > 0.001).

TABLE V. ANOVA FOR EACH DATASET OF THE ALGORITHM

| Datasets | Source | DF | Sum of squares | Mean Square | F | Signature |
|---|---|---|---|---|---|---|
| **October, 2022** | Between groups | 9 | 5515.022 | 612.780 | 0.690 | **0.718** |
| | Within groups | 347 | 308235.712 | 888.287 | | |
| | Total | 356 | 313750.734 | | | |
| **March, 2023** | Between groups | 9 | 1066.636 | 118.515 | 0.171 | **0.997** |
| | Within groups | 480 | 332711.768 | 693.150 | | |
| | Total | 489 | 333778.403 | | | |
| **October, 2023** | Between groups | 9 | 951.823 | 105.758 | 0.264 | **0.984** |
| | Within groups | 463 | 185482.213 | 400.610 | | |
| | Total | 472 | 186434.036 | | | |

TABLE VI. SESSION GAP AND CONSECUTIVE HOURS (IN A DAY) FOR ALL MAT/STA COURSES.

| Semesters | October, 2022 | | March, 2023 | | October, 2023 | |
|---|---|---|---|---|---|---|
| Stages | Act. | Imp. | Act. | Imp. | Act. | Imp. |
| Minimum Session Gap | 6 | 9 | 3 | 9 | 4 | 19 |
| Average Session Gap | 18 | 13 | 18 | 22 | 10 | 21 |
| Average Consecutive hours in a day | 7 | 4 | 6 | 4 | 7 | 4 |

Act. = Actual; Imp. = Improved

Session gaps (see Table VI) improved significantly, with minimum gaps increasing to 3 timeslots in October 2022, 6 in March 2023, and 15 in October 2023, indicating that 2-hour sessions are spread across 1–2 days. Reducing daily consecutive hours further contributed to more efficient timetables across all three semesters. Consequently, it provides a more balanced distribution of morning and afternoon timeslots across all groups' timetables (see Fig. 7).

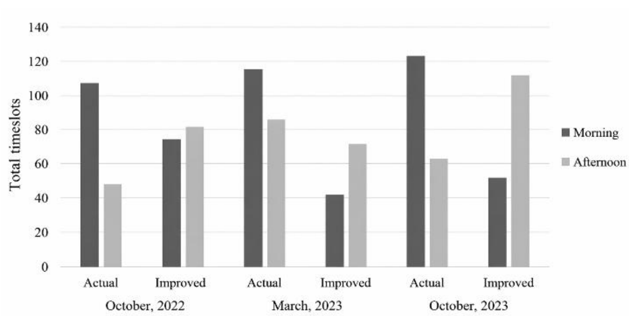


Fig. 7. Morning and afternoon timeslot distributions (all groups).

From the lecturer's timetable perspectives (Fig. 8), the distributions of morning and afternoon timeslots are more balanced in October 2022 and March 2023. In contrast, achieving this balance in October 2023 is difficult due to the higher complexity of multiple groups with more MAT/STA courses and shared-teaching constraints.

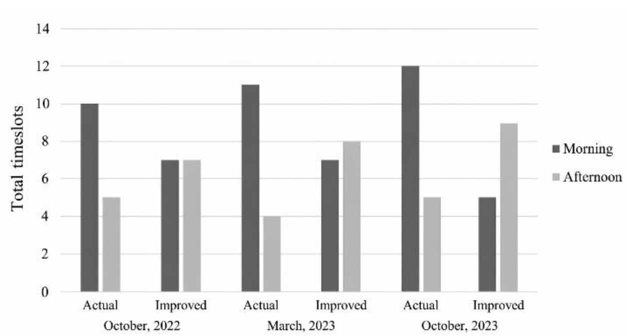


Fig. 8. Morning and afternoon timeslot distributions (all lecturers).

As a baseline comparison, 10 instances from ITC2002 and 12 from ITC2007 (Track 2) were evaluated. The results are compared with existing methods [2], [9], [17], which applied variants of evolutionary algorithms to the same benchmark datasets. The performance of benchmark datasets is measured by distance to feasibility (number of unallocated events) and the penalty cost of soft constraint violations. The soft constraints are included as follows:

- Students should not be scheduled to attend an event in the last timeslot of a day (that is, timeslots 9, 18, 27, 36, or 45).
- Students should not have to attend three (or more) events in consecutive timeslots occurring on the same day.
- Students should not be required to attend only one event on a particular day.

For the ITC2002 dataset, certain constraints are relaxed due to its slightly different characteristics, including session gaps, precedence events, and available timeslots. The proposed algorithm produces feasible solutions with no unallocated events for all instances (see Table VII). However, the resulting soft-constraint penalty costs are less competitive than those of previous works (see Table VIII).

TABLE VII. COMPARISON OF UNALLOCATED COURSES ON ITC2002 BENCHMARKS.

| Instance | Proposed EA | [17] | [2] |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 |

TABLE VIII. COMPARISON OF PENALTY COST ON ITC2002 BENCHMARKS

| Instance | Proposed EA | | | | [12] | [2] |
|---|---|---|---|---|---|---|
| | *Initial Avg.* | *Imp. Avg.* | *Initial Best* | *Imp. Best* | | |
| 1 | 599 | 444 | 566 | 407 | 52 | 77 |
| 2 | 585 | 443 | 544 | 405 | 20 | 250 |
| 3 | 568 | 534 | 540 | 512 | 78 | 100 |
| 4 | 831 | 674 | 783 | 611 | 74 | 130 |
| 5 | 922 | 806 | 842 | 704 | 71 | 326 |
| 6 | 886 | 767 | 837 | 681 | 6 | 359 |
| 7 | 1234 | 1114 | 1159 | 1049 | 6 | 574 |
| 8 | 680 | 545 | 628 | 490 | 15 | 383 |
| 9 | 630 | 507 | 586 | 472 | 32 | 256 |
| 10 | 589 | 499 | 516 | 460 | 58 | 73 |

Avg. = Average; Imp. = Improved

Meanwhile, constraints for ITC2007 (Track 2) are more complex than those for ITC2002, as events are allocated to specific time slots only. In addition, there is a precedence requirement that certain events should occur before certain others. Nevertheless, the session gaps are also relaxed during testing of the ITC2007 (Track 2) instance. Consequently, only three instances (3, 7, and 8) can achieve zero unallocated events by the proposed algorithm (see Table IX).

TABLE IX. COMPARISON OF UNALLOCATED COURSES ON ITC2007 (TRACK 2) BENCHMARKS.

| Instance | Proposed EA | [2] | [9] |
|---|---|---|---|
| 1 | 18 | 0 | 0 |
| 2 | 37 | 0 | 0 |
| **3** | **0** | **0** | **0** |
| 4 | 2 | 0 | 0 |
| 5 | 34 | 0 | 0 |
| 6 | 24 | 0 | 0 |
| **7** | **0** | **0** | **0** |
| **8** | **0** | **0** | **0** |
| 9 | 34 | 0 | 0 |
| 10 | 26 | 0 | 0 |
| 11 | 5 | 0 | 0 |
| 12 | 10 | 0 | 0 |

Furthermore, Table X presents a comparison of soft-constraint violations (penalty costs) for ITC2007 (Track 2). The penalty costs for initialization and improvement (Swapping 1 and Swapping 2) are compared with those in previous studies. The improved results consistently outperformed the initial solutions, demonstrating that the proposed variation operator significantly improves solution quality. Even though the proposed approach did not always outperform existing methods such as EM-GD [17], HH-PSO [2], and IPGALS [9], the observed improvements from the initial stage to the improved stage confirm that the algorithm can adapt and refine the solution effectively. This demonstrates robustness in maintaining quality, even at larger problem sizes.

TABLE X. COMPARISON OF PENALTY COST ON ITC2007 (TRACK 2) BENCHMARKS.

| Instance | Proposed EA | | | | [2] | [9] | [9] |
|---|---|---|---|---|---|---|---|
| | ***Initial Avg.*** | ***Imp. Avg.*** | ***Initial Best*** | ***Imp. Best*** | | ***Avg.*** | ***Best*** |
| 1 | 974.86 | 735.9 | 822 | **417** | 70 | 437 | **409** |
| 2 | 1073.6 | 857.35 | 1011 | 813 | 50 | 397 | 381 |
| 3 | 2544.55 | 1084.1 | 2356 | 996 | 150 | 214 | 195 |
| 4 | 2393.75 | 1108.2 | 2110 | 992 | 228 | 242 | 211 |
| 5 | 845.5 | 455 | 792 | 424 | 4 | 0 | 0 |
| 6 | 1028.55 | 387.05 | 986 | 353 | 0 | 0 | 0 |
| 7 | 1778.7 | 348.85 | 1682 | 320 | 0 | 0 | 0 |
| 8 | 2179.25 | 329.95 | 2095 | 300 | 0 | 0 | 0 |
| 9 | 1011.55 | 863.55 | 927 | **798** | **1794** | 0 | 0 |
| 10 | 1067.8 | 795.9 | 992 | 750 | 160 | 654 | 476 |
| 11 | 2155.4 | 1150.85 | 1935 | 1104 | 230 | 149 | 135 |
| 12 | 2144.65 | 1115 | 1916 | 1023 | 34 | 169 | 153 |

Avg. = Average; Imp. = Improved

## IV. CONCLUSION

In this study, an evolutionary algorithm with a dynamic local operator was used to allocate MAT/STA courses within the Mathematical Sciences Department at UiTM Sarawak Branch. The dynamic local operator was used to relocate and recompute the session gaps for 2+2 lecture hours of MAT/STA courses while satisfying a balanced distribution within a pre-scheduled timetable (limited available time slots).

Computational results are compared with the manual solutions for three semesters. It shows that the proposed algorithm can handle more complex scenarios (multiple groups with more MAT/STA courses) at the cost of a later end time. The proposed variation operator significantly improves solution quality, even when a good initial population has already been generated (compared to manual solutions). On average, the penalty value is reduced by up to 66.30%, and up to 15 timeslots improve the session gap. The proposed algorithm was further tested on ITC2002 and ITC2007 (Track 2) to serve as a baseline for existing methods. With certain constraint relaxations, the benchmark dataset results indicate that the proposed algorithm maintains solution quality even in larger, more complex settings.

## ACKNOWLEDGMENT

This research was partially funded by the Universiti Malaysia Sarawak (UNIMAS).

## REFERENCES

[1] S. Imran Hossain, M. A. H. Akhand, M. I. R. Shuvo, N. Siddique, and H. Adeli, “Optimization of University Course Scheduling Problem using Particle Swarm Optimization with Selective Search,” Expert Syst. Appl., vol. 127, pp. 9–24, Aug. 2019, doi: 10.1016/j.eswa.2019.02.026.

[2] Z. Iqbal, R. Ilyas, H. Y. Chan, and N. Ahmed, “Effective Solution of University Course Timetabling using Particle Swarm Optimizer based Hyper Heuristic approach,” Baghdad Science Journal, vol. 18, no. 4, pp. 1465–1475, 2021, doi: 10.21123/bsj.2021.18.4(Suppl.).1465.

[3] T. Thepphakorn, S. Sooncharoen, and P. Pongcharoen, “Particle Swarm Optimisation Variants and Its Hybridisation Ratios for Generating Cost-Effective Educational Course Timetables,” SN Comput. Sci., vol. 2, no. 4, Jul. 2021, doi: 10.1007/s42979-021-00652-2.

[4] I. A. Abduljabbar and S. M. Abdullah, “An evolutionary algorithm for solving academic courses timetable scheduling problem,” Baghdad Science Journal, vol. 19, no. 2, pp. 399–408, 2022, doi: 10.21123/BSJ.2022.19.2.0399.

[5] E. C. Perez, O. M. Rios, D. P. Bautista, S. S. Sanchez, and F. A. Acevedo, “A Genetic Algorithm Solution for Scheduling Problem,” in CONIIN 2021 - 17th International Engineering Congress, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/CONIIN54356.2021.9634725.

[6] J. S. Tan, S. L. Goh, S. Sura, G. Kendall, and N. R. Sabar, “Hybrid particle swarm optimization with particle elimination for the high school timetabling problem,” Evol. Intell., vol. 14, no. 4, pp. 1915–1930, Dec. 2021, doi: 10.1007/s12065-020-00473-x.

[7] C. Altıntaş and T. Yiğit, “Analysing the effects of classroom utilisation with a self-generating multimeme memetic algorithm for the exam timetabling problem,” Advances in Artificial Intelligence Research (AAIR), vol. 1, no. 2, pp. 43–51, 2021, [Online]. Available: https://dergipark.org.tr/tr/pub/aair

[8] D. F. Dofadar, R. H. Khan, S. Hasan, T. A. Taj, A. Shakil, and M. Majumdar, “A Hybrid Evolutionary Approach to Solve University Course Allocation Problem,” in Proceedings - 2021 International Conference on Artificial Intelligence and Blockchain Technology, AIBT 2021, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 48–52. doi: 10.1109/AIBT53261.2021.00015.

[9] A. Rezaeipanah, S. S. Matoori, and G. Ahmadi, “A hybrid algorithm for the university course timetabling problem using the improved parallel genetic algorithm and local search,” Applied Intelligence, vol. 51, no. 1, pp. 467–492, Jan. 2021, doi: 10.1007/s10489-020-01833-x.

[10] G. K. Edmund K. Burke, Search Methodologies. Boston, MA: Springer US, 2014. doi: 10.1007/978-1-4614-6940-7.

[11] A. Hamdan, S. San Nah, G. Say Leng, C. Kang Leng, and T. Wei King, “Recent Evolutionary Algorithm Variants for Combinatorial Optimization Problem,” Applications of Modelling and Simulation, vol. 7, pp. 214–238, Dec. 2023, [Online]. Available: https://arqiipubl.com/ojs/index.php/AMS_Journal/article/view/515

[12] R. Lewis, “A survey of metaheuristic-based techniques for University Timetabling problems,” OR Spectrum, vol. 30, no. 1, pp. 167–190, Jan. 2008, doi: 10.1007/s00291-007-0097-0.

[13] R. Lewis, “A time-dependent metaheuristic algorithm for post enrolment-based course timetabling,” Ann. Oper. Res., vol. 194, no. 1, pp. 273–289, Apr. 2012, doi: 10.1007/s10479-010-0696-z.

[14] H. Cambazard, E. Hebrard, B. O’Sullivan, and A. Papadopoulos, “Local search and constraint programming for the post enrolment-based course timetabling problem,” Ann. Oper. Res., vol. 194, no. 1, pp. 111–135, Apr. 2012, doi: 10.1007/s10479-010-0737-7.

[15] C. Nothegger, A. Mayer, A. Chwatal, and G. R. Raidl, “Solving the post enrolment course timetabling problem by ant colony optimization,” Ann. Oper. Res., vol. 194, no. 1, pp. 325–339, Apr. 2012, doi: 10.1007/s10479-012-1078-5.

[16] G. Kobeaga, M. Merino, and J. A. Lozano, “An efficient evolutionary algorithm for the orienteering problem,” *Comput. Oper. Res.*, vol. 90, pp. 42–59, Feb. 2018, doi: 10.1016/j.cor.2017.09.003.

[17] S. Abdullah, H. Turabieh, B. McCollum, and P. McMullan, “A hybrid metaheuristic approach to the university course timetabling problem,” Journal of Heuristics, vol. 18, no. 1, pp. 1–23, Feb. 2012, doi: 10.1007/s10732-010-9154-y.