OMU: A Multi-Swarm Particle Swarm Optimization Solution for the University Course Timetabling Problem

Gian Myrl D. Renomeron
CMSC 199.1: Research in Computer Science I
Division of Natural Sciences and Mathematics
University of the Philippines Tacloban College
gdrenomeron@up.edu.ph

OMU: A Multi-Swarm Particle Swarm Optimization Solution for the University Course Timetabling Problem

1 Introduction

Academic institutions face the challenging task of effective resource management in their administrations without compromising the quality of the educational experience in this dynamic world of academia. [1] Among the most complicated logistical problems that universities encounter is the so-called *University Course Timetabling Problem* (UCTP). [2] [3] This problem deals with course-instructor-student assignment to particular time slots and rooms. It includes many constraints such as room capacities, faculty availability, and course prerequisites. [4] For solving these issues and establishing a standardized benchmark, the Second International Timetabling Competition (ITC-2007) proposed Track 3: Curriculum-Based Course Timetabling (CB-CTT). [5] This track defines the UCTP as the assignment of lectures to periods and rooms, subject to hard constraints that must be strictly satisfied (e.g., no room clashes) and soft constraints aimed at optimizing resource utilization and minimizing inconvenience (e.g., spreading lectures evenly). ITC-2007 provided datasets that have since become a critical benchmark for evaluating and comparing optimization algorithms for course timetabling. Many optimization techniques and algorithms have been applied to solve the CB-CTT problems. Some of the developed algorithms include heuristics, such as integer programming [6] [7] and metaheuristics-based algorithms, including local search based and hybrid algorithms. [8] [9] [10], [11], [12], [13], [14], [15] The ideal schedule remains a challenge to prepare, especially for larger institutions, because manual methods lead to conflicts and inefficiencies.

Latest advanced computational approaches emerged to address these problems, and among them *Swarm Intelligence* (SI) is one of the most promising directions. [16] Among the algorithms of this group, there is an attention-grabber - *Particle Swarm Optimization* (PSO), distinguished by adaptability and efficiency in searching large solution spaces. [17] [18]. The algorithm is represented by a set of particles, which are possible solutions; each particle moves in the search space, and its position changes according to individual and collective experiences.[19] [20] Still, despite many strengths, PSO fails at times to solve problems efficiently within UCTP, often failing to find the best solution under tight constraints. [3]

Multi-Swarm Optimization (MSO) is designed to split up the main swarm into smaller, specialized sub-swarms that concurrently operate in exploiting different regions of the solution space simultaneously. [21] [22] It aims to improve the capabilities of PSO by focusing on issues such as search diversity and the global optimizing ability of the algorithm. [23] One of such particular modifications of the above-mentioned technique is MPSO: *Multi-Swarm Particle Swarm Optimization*, which dynamically varies the sub-swarms in the standard PSO in both exploration and exploitation for a good performance to provide capabili-

ties of global and local search. [22] [24] [23] To make this approach balanced, mechanisms for exploration and exploitation are integrated appropriately such that different solution domains would be explored in detail while optimizing the potential solutions in a balanced manner. [25]

This paper introduces a new approach to solving the UCTP based on Multi-swarm PSO. MPSO is specially designed for university course timetabling with diverse constraints, while search methods are efficient in finding optimal timetables. This approach promises to provide schedules that are conflict-free as well as resource-efficient and meet the specific needs of educational institutions. This effort aims to contribute towards scalable and realistic automation and optimization of timetabling, responding to one of the most complex problems in academic administration.

1.1 Background of the Study

University Course Timetabling Problem (UCTP)

The *University Course Timetabling Problem* (UCTP) is defined as a problem of making assignments for courses, instructors, and students to particular time slots and rooms according to numerous constraints, for example room capacities and instructor availability.[1] [3] [17] UCTP is a classic problem of academic optimization, and quality methods are demanded for efficient scheduling.[2] [26] [27]

Heuristic Algorithms

Heuristic Algorithms are heuristic methods to solve optimization problems that should find acceptable solutions within reasonable time. Problem-specific rules or strategies will be employed to search effectively in the solution space to provide a good enough solution rather than guaranteeing optimality in the whole problem space. For example, greedy algorithms, local search, and evolutionary heuristics have been very popular for applications in scheduling problems like UCTP [1] [26].

Swarm Intelligence

Swarm Intelligence refers to bio-inspired algorithms based on the collective behavior of animals such as ants, bees, or birds. [28] [29] It is applied to optimization problems because it can efficiently and adaptively search very large solution spaces. [3] [28]

Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is the best known among SI algorithms. [30] It simulates the motions of particles, which denote solutions, in the search space and updates them based on individual and collective experiences. [18] [31] PSO performs very well on many

optimization problems, such as UCTP, but still struggles with tricky constraints. [17] [3]

Multi-Swarm Optimization (MSO)

Multi-Swarm Optimization (MSO) is an advanced extension of swarm intelligence algorithms. The basic idea in MSO is to divide the primal swarm into multiple sub-swarms that scan concurrently the different areas of the solution space. A concurrent search makes the algorithm have a better opportunity to discover global optima with effective balancing between exploration and exploitation.[21]. [22]

Multi-swarm Particle Swarm Optimization (MPSO)

Multi-Swarm Particle Swarm Optimization (MPSO) is a PSO algorithm developed to overcome the lack of diversity of search and local optima avoidance in the process of complex optimization problems, by splitting the main swarm into a considerable number of subswarms that explore different regions in parallel with the solution space, hence, it promotes diversity and the likelihood of finding the global optima. [23] [24] [32] The difference gives the algorithm an excellent improvement in searching for such solutions for complex problems, such as the UCTP, which balanced exploration and exploitation would provide.

1.2 Related Works

In the field of UCTP, heuristic approaches have been studied for extensive potential towards providing valid and effective solutions with respect to constraints. Metaheuristics aim at establishing quick but effective solutions by approximating optimal schedules rather than executing exhaustive searches, which can sometimes be very computationally expensive.

A two-stage very popular heuristic strategy by Bong Chia Lih et al. [26] gives the first stage to be the grouping of all courses that can be held at the same time, and the second stage to be the assignment of time slots and venues for these groups. Meanwhile, the approach lowers the complexity of this problem, and it has already been successfully applied using real university data and it can handle real-world constraints very efficiently. Similar to Zhang et al., [1] an efficient greedy metaheuristic algorithm handles room and time-slot assignments. Thus, the presented approach, emphasizing the predefined constraints, can meet the required flexibility by these timetables and institutional requirements.

Other metaheuristic methods applied in solving UCTP include Particle Swarm Optimization (PSO). PSO is another optimization technique that is increasingly gaining acceptance in solving UCTP problems. The concept was derived from the social behavior a swarm of birds or of fish follows. This makes it possible to adopt the efficient exploration of large

search spaces. Oswald and Durai [3] proposed a hybrid PSO with strategies for improving search applied particularly for UCTP. This was well balanced between exploration and exploitation so the robust timetabling solutions in reference to Oswald et al 2013. Similarly, Chen and Shih [17] created a constriction PSO model integrated with local search strategies to enhance both the gain in generation speed of solutions and the quality of the solution obtained. It has shown the effectiveness of PSO in solving challenging scheduling problems.

Multi-Swarm Particle Swarm Optimization (MPSO) is an advanced variant of the standard PSO algorithm, aimed at enhancing performance in dynamic and multimodal optimization problems by dividing the population into multiple sub-swarms. Tim Blackwell and Jürgen Branke, in multiswarm PSO, incorporate mechanisms appropriate to dynamic optimization environments. [24] This methodology, while trying to maintain diversity over multiple swarms, focuses on keeping different swarms robustly tracking changing optima. It incorporates exclusion where swarms do not collapse to the same peak, as each pair of swarms within a predefined exclusion radius have weaker swarms reinitialized. It helps different swarms in this way to explore the different regions of the search space. In addition, anti-convergence occurs when all swarms converge; the least-fit swarm is reinitialized to search for emerging or unexplored peaks, ensuring adaptability in changing environments [24]. This algorithm further incorporates quantum particles which are randomly dispersed within a determined distance of the best known position inside the swarm. These maintain intraswarm diversity by facilitating the ability to quickly react and follow shifts in peeks. Combining these concepts permits MPSO to balance global searches and local explorations efficiently with its performance surpassing PSO over dynamic multimodal landscapes [33].

Generalizing the metaheuristic methods embraced here, such as greedy algorithms, adaptive genetic strategies, and PSO-based techniques, indicate that these are feasible for solving UCTP. In fact, these methods are not a global optimality guarantee; however, speed and flexibility in an ideally minimal number of computations make them highly valuable tools for educational institutions, especially for large-scale tasks where a large number of constraints are to be balanced.

2 Statement of the Problem

The UCTP problem is quite complex and key for any institution of learning because it considers assigning courses to specific rooms and time slots while considering constraints such as room capacities, course prerequisites, and faculty courses. [2] To this date, most universities still consider non-optimal manual or heuristic methods, and the inefficiencies include uneven teaching loads and a mismatch between qualifications and requirements in course teaching. [3] [20] While the complexity of the problem is increased, it happens to be highly infea-

sible to find the best solutions, especially when there are conflicting constraints that need to be fulfilled.

The existing work [3] [18] [17] used PSO-based techniques in solving UCTP but highly developed approaches are required to be found due to the complexity of the problem, such as MPSO, this improves the original PSO technique by splitting the swarm into many sub-swarms that simultaneously start exploring different regions of solution space. This allows for a higher degree of diversity in search and avoids getting trapped in the local optimum for a better possibility of finding the global optimum.

This paper focuses on the development of a solution for UCTP using MPSO, that is, to explore the vast solution space with much greater efficiency than before, paying attention to constraints like room capacities, course prerequisites, and scheduling conflicts. In terms of metrics like balance of workloads, course-faculty alignment, and computational efficiency, this proposed approach is supposed to bring about improvements in quality and fairness in the courses scheduled within institutions.

3 Objectives of the Study

3.1 General Objective

To design and develop a Multi-Swarm Particle Swarm Optimization (MPSO) algorithm to solve the University Course Timetabling Problem (UCTP).

3.2 Specific Objectives

1. To develop and analyze the MPSO algorithmic usage and efficacy in solving UCTP.
2. To model the UCTP with crucial characteristics like room availability, course prerequisites, and faculty courses within the framework of MPSO.
3. To perform a state-of-the-art analysis by comparing with other existing approaches to solve the UCTP

4 Scope and Limitation

This paper introduces a new optimization approach from the perspective of MPSO for the UCTP. It examines some of the critical constraints in intense scrutiny and in depth, like room availability, course prerequisites, and faculty course assignments. The MPSO-based model aims to improve the efficiency of schedules across several constraints within institutions. In order to ensure reliability and adaptability in various contexts and environments, real-time validation is carried out on the model. Lastly, the evaluation of the MPSO model will also show comparison with existing algorithms in order to meet the preferences of faculties as well as serve as a more holistic benchmark.

However, some limitations have been encountered in this study. Even though the system is targeting to make resource usage up to the fullest as well as making the conflict avoidance probability to be at minimum, it does not comprise those dynamic changes which exist on the process of scheduling such as changing the courses at the last minute or rooms available suddenly. The model will be tested strictly within predefined constraints and data originating from multiple institutions, which can limit the general applicability of obtained results in different educational contexts. Therefore, although the study is focused on satisfying hard constraints, the subjective aspects of scheduling-for example, preferences of faculty members-might not be fulfilled fully by the final solution, but they will be taken into account for benchmarking purposes.

5 Significance of the Study

The importance of this paper is developing a new approach for solving the University Course Timetabling Problem using Multi-Swarm Particle Swarm Optimization. To be more specific, because numerous methods have been widely applied to solve UCTP, MSPSO is the first approach that explores more extensive areas of solutions through the use of more than one sub-swarm. This new technique aims to generate high-quality, conflict-free timetables that respect necessary constraints such as room capacities, course prerequisites, and faculty schedules, thereby advancing the field of university scheduling.

Besides the academic value, this research opens up possibilities for wider applications of MSPSO in fields requiring efficient scheduling and resource management. With the establishment of MSPSO as an efficient strategy for UCTP, this study opens avenues for its use in other complex scheduling scenarios related to healthcare, transportation, and manufacturing. These contexts are critical where optimum planning and resource utilization must occur. The research outcomes from this study might stimulate new studies and further applications in MSPSO for diverse industry fields.

Therefore, this paper not only develops a novel solution for scheduling at the university level but also lays down a foundation that may help in further studying MSPSO as a broad tool for automated scheduling in multiple industries.

6 Theoretical and Conceptual Framework

The theoretical and conceptual framework of this study consists of five separate yet connected components: Data Collection, Data Processing, Approach Construction, Approach Evaluation, and Deployment. The workflow starts with the Data Collection phase, collecting benchmark datasets and optimization problem instances. These datasets usually are publicly available and typically represented in text or XML formats. This phase ensures that there is adequate sourcing and data variety to support the solid analysis of the optimization model against different conditions. Steps undertaken in the Data Processing step include cleaning the data to remove inconsistencies and make sure that all

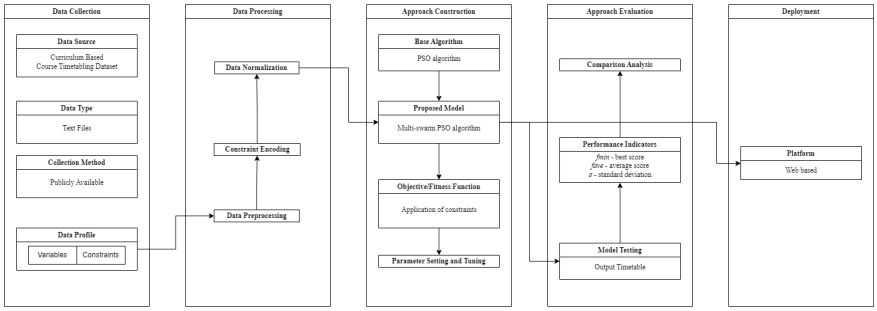

Figure 1: Theoretical and Conceptual Framework

inconsistencies in constraints are encoded in apt forms of mathematics for solving before preparing the data for optimized use. Data cleaning removes errors or missing values that could have otherwise affected the model's performance while encoding constraints take problem-specific rules and transform them into a format that could be used by the model. Normalization and data categorization are also considered uniform since the optimization model can cope well with the data. All these processes, therefore, collectively provide standardized and reliable input for the optimization phase. This ensures that the information is valid and appropriate to the model in question. It will act as a foundation for later phases.

The construction of the approach deals with the creation of the model of multi-swarm optimization. This acts as a core method for the optimization problem solution. A part of this component deals with the base model setup and setting parameters that make the model capable of appropriately addressing the needs of the problem. The Approach Evaluation phase involves testing the performance of the model based on whether it satisfies any required objectives. This phase must be done to determine how well the proposed model is and make adjustments according to what is needed to increase its applicability. Lastly, the solution would be deployed by presenting an optimized solution via the use of a web-based interface, where users will have the ability to upload their problem instances and find real-time optimized outputs for their respective problems. All parts are included to be effective contributors to the solution development process, ensuring that data is handled properly, models built and evaluated appropriately, and the final solution is practically deployable.

6.1 Concepts, Theories, and Methodologies Reused

This paper employs current swarm intelligence theories and algorithms, mainly in Particle Swarm Optimization (PSO) [19] and Multi-Swarm Optimization (MSO) [22] [24]. PSO is an established algorithm to efficiently search large solution spaces by a set of particles that are exploring a search area with indi-

vidual and collective experiences. Parsopoulos and Vrahatis [34] first introduced neighborhood-based swarming in global and local variants in the introduction of the MSO approach. That paper set a solid basis for multi-swarm frameworks. The structure of the neighborhood and dynamic tuning of parameters discussed there provided important inspirations for later developments in multi-swarm optimization. One of the first spectacular applications of these multi-swarm approaches is given by Blackwell and Branke [22], who studied MSO in dynamic settings. These methodologies are crucial to establish the foundation for MPSO.

6.2 Concepts, Theories, and Methodologies Modified

To improve existing PSO and MSO methodologies, the research here modifies the base methodologies to take on a multi-swarm structure particularly designed for UCTP, called MPSO. The MPSO algorithm divides the main swarm into smaller, specialized sub-swarms operating in parallel to explore the solution space more effectively at different parts. [23] [22] [24] The improvement is designed to circumvent the limitations of lack of search diversity and trapping into local optima through the encouragement of a greater exploration of solution possibilities along with the exploitation of promising regions. The modification improves better adaptability and efficiency in generating feasible course timetables.

6.3 Novel Concepts, Theories, and Methodologies Introduced

This research introduces a new optimization approach for UCTP via MPSO, specifically designed to address complex timetabling constraints such as room availability, faculty preferences, and scheduling conflicts. Its novelty lies in the combination of dynamic swarm partitioning and targeted exploration, allowing the MPSO algorithm to achieve workload balance and enhance resource use within educational institutions [21]. The proposed MPSO-based methodology balances mechanisms for exploration and exploitation, ensuring that the different solution domains are discovered thoroughly to allow the provision of optimal scheduling solutions.

7 Methodology

7.1 Data Collection

The dataset used in this research is based on the ITC2007 Track 3: Curriculum-Based Course Timetabling (CB-CTT). This dataset has instances where it contains detailed information. This information include a header and four main sections: courses, rooms, curricula, and constraints, with each section containing arrays corresponding to a specific component in the problem and all scalar values summarised in the header. Each dataset instance provides the basic information for testing and validating the proposed approach.

```
Name: ToyExample
Courses: 4
Rooms: 2
Days: 5
Periods_per_day: 4
Curricula: 2
Constraints: 8

COURSES:
SceCosC Ocra 3 3 30
ArcTec Indaco 3 2 42
TecCos Rosa 5 4 40
Geotec Scarlatti 5 4 18

ROOMS:
A 32
B 50

CURRICULA:
Cur1 3 SceCosC ArcTec TecCos
Cur2 2 TecCos Geotec

UNAVAILABILITY_CONSTRAINTS:
TecCos 2 0
TecCos 2 1
TecCos 3 2
TecCos 3 3
ArcTec 4 0
ArcTec 4 1
ArcTec 4 2
ArcTec 4 3

END.
```

Figure 2: Example of CB-CTT Instance

The header describes the important properties of the instance: name of the instance, number of courses, rooms, working days, periods per day, curricula, and constraints. These values determine the scope and size of the timetabling problem and the boundaries of feasible scheduling.

In the **Courses** section, every course is specified to include the assigned teacher, number of lectures required, minimum number of working days that lectures should be spread, and number of students enrolled.

The **Rooms** section defines the rooms available for scheduling along with their seating capacities.

Under **Curricula**, the courses of similar curricula are combined together such that no two classes from a particular curriculum fall at the same time.

The **Unavailability Constraints** section specifies day-period combinations during which certain courses cannot be scheduled. These constraints typically arise from limitations such as teacher unavailability or room restrictions.

This structured dataset gives a stable base for testing and verification of the proposed MPSO, thus ensuring that all the constraints imposed on it are well described.

7.2 Data Processing

The data processing phase transforms the raw dataset into a structured and optimized format suitable for integration into the MPSO framework. All problem components and constraints are accurately represented to allow seamless initialization of particles and candidate solution evaluation.

The dataset contains courses, rooms, curricula, and unavailability constraints, which are parsed and normalized in order to achieve consistency. Time-related data, that is, days and periods, are converted to numerical indices, and the constraints are encoded to lead the process of optimization. The parsed data is structured into a particle representation, that is, encoding each candidate solution of the timetabling problem as a list of scheduling entries in the format:

$$(\text{Day}, \text{Period}, \text{Room}, \text{Course}).$$

Constraint Encoding

The constraints, central to the problem, can be categorized into two types, namely hard constraints and soft constraints. Hard constraints are mandatory and must be strictly satisfied, while soft constraints are desirable properties that contribute to the overall quality of the solution. The details and mathematical representations of these constraints are given as follows:

Symbol	Definition
d, p, r, c	Day, Period, Room, and Course respectively.
S	Set of all scheduling entries, where each entry is (d, p, r, c).
U	Set of unavailability constraints, specifying unavailable periods for certain courses.
$\text{Curr}(k)$	Set of courses in curriculum k.
$T(c)$	Teacher assigned to course c.
$\text{students}(c)$	Number of students enrolled in course c.
$\text{capacity}(r)$	Capacity of room r.
$\text{min\_days}(c)$	Minimum number of days required for course c.
R_c	Set of rooms assigned to course c across all scheduled periods.

Table 1: Symbols used for the CB-CTT problem

Hard Constraints

1. **Lecture Assignment**: Each lecture of a course must be assigned to a unique combination of day, period, and room. Overlaps are strictly prohibited.

$$\forall (d, p, r, c_i), (d', p', r', c_j) \in S, \quad i \neq j \Rightarrow (d, p, r) \neq (d', p', r').$$

2. **Room Occupancy**: No two lectures can occupy the same room at the same time.

$$\forall (d, p, r, c_i), (d, p, r, c_j) \in S, \quad i \neq j \Rightarrow c_i \neq c_j.$$

3. **Conflicts**: Courses in the same curriculum or taught by the same teacher must not overlap in time.

$$\forall (d, p, r, c_i), (d, p, r', c_j) \in S, \quad (c_i \in \text{Curr}(k) \vee T(c_i) = T(c_j)) \Rightarrow p_i \neq p_j.$$

   Here, $\text{Curr}(k)$ represents the courses in curriculum k, and $T(c)$ is the teacher assigned to course c.

4. **Availability**: Courses cannot be scheduled during unavailable periods as specified in the dataset.

$$\forall (c, d, p) \in U, \quad (d', p', r, c') \in S, \quad c' = c \Rightarrow (d' \neq d \vee p' \neq p).$$

Soft Constraints

1. **Room Capacity**: Assigning a course to a room with insufficient capacity incurs a penalty proportional to the seating deficit.

$$P_1 = \sum_{(d,p,r,c) \in S} \max(0, \text{students}(c) - \text{capacity}(r)).$$

2. **Minimum Working Days**: Scheduling a course for fewer days than its minimum requirement results in a penalty.

$$P_2 = \sum_{c \in C} 5 \cdot \max(0, \text{min\_days}(c) - |\{d \mid (d, p, r, c) \in S\}|).$$

3. **Room Stability**: Assigning a course to multiple rooms across its schedule incurs a penalty for each additional room used.

$$P_3 = \sum_{c \in C} \max(0, |R_c| - 1), \quad R_c = \{r \mid (d, p, r, c) \in S\}.$$

4. **Curriculum Compactness**: Isolated lectures for a curriculum within a day are penalized, promoting schedule compactness.

$$P_4 = \sum_{k} \sum_{d} \sum_{p \in P_d} \text{is\_isolated}(p),$$

where $\text{is\_isolated}(p) = 2$ if no adjacent lectures exist for curriculum k.

Fitness Function

The overall fitness function integrates the penalties for soft constraints and serves as the objective function for optimization. It is expressed as:

$$\text{Fitness} = P_1 + P_2 + P_3 + P_4.$$

Validation and Testing

The parsed data is verified for completeness and consistency during the process. It checks to ensure that all courses, rooms, and curricula have been represented accurately and the constraints are correctly encoded. Thus, the structured and validated data forms a robust base for initializing particles and guiding the optimization process.

The approach of comprehensive data processing will prepare the problem constraints and components for the MPSO framework in an accurate manner so that hard constraints are maintained and soft constraint penalties are optimized.

7.3 Approach Construction

The approach begins with initializing particles in the Multi-Swarm Particle Swarm Optimization (MPSO) algorithm. Each particle represents a potential solution—a timetabling configuration. 1. **Initialization**: Particles are initialized using heuristic methods like Largest Degree (LD), ensuring an initial feasible solution that respects hard constraints. 2. **Optimization Loop**: The MPSO iteratively improves solutions by updating particle positions and velocities based on personal best and global best configurations. 3. **Diversity

Mechanisms**: The algorithm employs exclusion, anti-convergence, and quantum reinitialization mechanisms to maintain diversity and prevent premature convergence. 4. **Fitness Evaluation**: Each particle's fitness is computed based on a composite objective function incorporating penalties for violations of both hard and soft constraints.

7.4 Approach Evaluation

The performance of the MPSO algorithm is validated against the ITC2007 datasets. The evaluation process includes: - **Validation Tool**: A provided executable from ITC2007 is used to evaluate the feasibility and quality of the generated timetable. - **Metrics**: Hard constraint violations are strictly penalized, while soft constraint violations contribute to the overall cost. The fitness function aims to minimize this cost. - **Comparison**: The proposed algorithm is benchmarked against existing methods to highlight its effectiveness in generating high-quality timetables.

7.5 Deployment

The final solution is implemented as a web-based application using the Streamlit Python framework. This interface allows users to upload dataset files, run the MPSO algorithm, and generate optimized timetables. The application outputs the timetable in a user-friendly format and provides a summary of the solution's quality, including constraint violations and overall cost.

8 Schedule of Activities

OBJECTIVES	**TARGET ACTIVITIES**	**TARGET ACCOMPLISHMENTS** (quantify, if possible)	**YEAR 1**											
			1	2	3	4	5	6	7	8	9	10	11	12

Objective 1	To examine how the MSPSO algorithm is used and how well it solves the UCTP.	1. Gather relevant literature on MSPSO applications. 2. Analyze the performance of MSPSO on test datasets. 3. Evaluate the effectiveness of MSPSO in solving UCTP with various constraints.													
Objective 2	To model the UCTP with crucial characteristics like room availability, course prerequisites, and faculty courses within the framework of MSPSO.	1. Identify and define critical constraints in UCTP. 2. Develop and model UCTP using MSPSO framework. 3. Test the model on synthetic and real data.													
Objective 3	To perform a state-of-the-art analysis by contrasting the MSPSO approach with current techniques used in the UCTP.	1. Benchmark MSPSO against other UCTP methods like GA and PSO. 2. Conduct performance evaluations and analysis.													

OBJECTIVES	**TARGET ACTIVITIES**	**TARGET ACCOMPLISHMENTS** (quantify, if possible)	**YEAR 2**											
			1	2	3	4	5	6	7	8	9	10	11	12
Objective 3	To perform a state-of-the-art analysis by contrasting the MSPSO approach with current techniques used in the UCTP.	1. Conduct a detailed analysis and comparison with existing UCTP techniques. 2. Publish results and make recommendations for improvements.												

References

[1] Dezhen Zhang et al. "A novel greedy heuristic algorithm for university course timetabling problem". In: (2014).

[2] Nancy Maribel Arratia-Martinez, Cristina Maya-Padron, and Paulina A Avila-Torres. "University course timetabling problem with professor assignment". en. In: *Math. Probl. Eng.* 2021 (Jan. 2021), pp. 1–9.

[3] Oswald C. and Anand Deva Durai C. "Novel hybrid PSO algorithms with search optimization strategies for a University Course Timetabling Problem". In: (Dec. 2013).

[4] Monica Torres, Kyla Kiela Villegas, and Maica Krizna Gavina. "Solving faculty-course allocation problem using integer programming model". en. In: *Philipp J. Sci.* 150.4 (June 2021), p. 679.

[5] ITC-2007 Organizers. "Curriculum Based Course Timetabling Problem Description for ITC-2007". In: (2007). URL: `http://www.cs.qub.ac.uk/itc2007`.

[6] G. Lach and M. Lübbecke. "Curriculum Based Course Timetabling: Optimal Solutions to the Udine Benchmark Instances". In: *Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (PATAT)*. Ed. by Edmund Burke and Michel Gendreau. 2008. URL: `http://patatconference.org/2008/`.

[7] E. Burke et al. "A Branch-and-Cut Procedure for the Udine Course Timetabling Problem". In: *Annals of Operations Research* (2008). Manuscript available at the time of publication.

[8] Asaju La'aro Bolaji et al. "An Improved Artificial Bee Colony for Course Timetabling". In: *Sixth International Conference on Bio-Inspired Computing: Theories and Applications* (2011), pp. 9–13. DOI: 10.1109/BICTA.2011.74.

[9] Tomáš Müller. "ITC2007 Solver Description: A Hybrid Approach". In: (2007). URL: http://www.unitime.org/itc2007.

[10] S. Abdullah et al. "An Investigation of a Genetic Algorithm and Sequential Local Search Approach for Curriculum-Based Course Timetabling Problems". In: (2010).

[11] M. Geiger. "Applying the Threshold Accepting Metaheuristic to Curriculum-Based Course Timetabling". In: *Annals of Operations Research* (2009), pp. 1–14.

[12] M. Geiger. "Multi-Criteria Curriculum-Based Course Timetabling: A Comparison of a Weighted Sum and a Reference Point Based Approach". In: *Evolutionary Multi-Criterion Optimization*. Springer, 2009, pp. 290–304.

[13] K. Shaker and S. Abdullah. "Incorporating Great Deluge Approach with Kempe Chain Neighbourhood Structure for Curriculum-Based Course Timetabling Problems". In: *Data Mining and Optimization, 2009. DMO'09. 2nd Conference on*. IEEE, 2009, pp. 149–153.

[14] Z. Lü, J. Hao, and F. Glover. "Neighborhood Analysis: A Case Study on Curriculum-Based Course Timetabling". In: *Journal of Heuristics* (2009), pp. 1–22.

[15] Z. Lü and J. Hao. "Adaptive Tabu Search for Course Timetabling". In: *European Journal of Operational Research* 200.1 (2010), pp. 235–244.

[16] H. Algethami and W. Laesanklang. "A Mathematical Model for Course Timetabling Problem With Faculty-Course Assignment Constraints". In: *IEEE Access* 9 (Aug. 2021), pp. 111666–111682. DOI: 10.1109/ACCESS.2021.3103495.

[17] Ruey-Maw Chen and Hsiao-Fang Shih. "Solving University Course Timetabling Problems Using Constriction Particle Swarm Optimization with Local Search". In: *Algorithms* 6 (Apr. 2013), pp. 227–244. DOI: 10.3390/a6020227.

[18] A I Ali and R Talal. "UCTP based on Hybrid PSO with Tabu Search Algorithm using Mosul university dataset". In: *International Journal of Computer Applications* 975 (2014).

[19] James Kennedy and Russell Eberhart. "Particle swarm optimization". In: *Proceedings of IEEE International Conference on Neural Networks* (1995), pp. 1942–1948.

[20] K. M. Ng Aldy Gunawan and H. L. Ong. "A Genetic Algorithm for the Teacher Assignment Problem for a University in Indonesia". In: *International Journal of Information and Management Sciences* 19.1 (Mar. 2008), pp. 1–16. DOI: 10.1000/ijims.2008.237502795.

[21] Nebojsa Bacanin et al. "Multi-Swarm Algorithm for Extreme Learning Machine Optimization". In: *Sensors* 22.11 (2022), p. 4204. DOI: 10.3390/s22114204. URL: https://doi.org/10.3390/s22114204.

[22] Jürgen Branke and Tim Blackwell. "Multi-swarm Optimization in Dynamic Environments". In: (2004). Ed. by Juan Julián Merelo Guervós et

al., pp. 307–317. DOI: 10.1007/978-3-540-24653-4_50. URL: https://link.springer.com/chapter/10.1007/978-3-540-24653-4_50.

[23] Xuewen Xia, Ling Gui, and Zhi-Hui Zhan. "A multi-swarm particle swarm optimization algorithm based on dynamical topology and purposeful detecting". In: *Applied Soft Computing* 67 (2018), pp. 126–140. ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2018.02.042. URL: https://www.sciencedirect.com/science/article/pii/S1568494618301017.

[24] T. Blackwell and J. Branke. "Multiswarms, exclusion, and anti-convergence in dynamic environments". In: *IEEE Transactions on Evolutionary Computation* 10.4 (2006), pp. 459–472. DOI: 10.1109/TEVC.2005.857074.

[25] Xi Wang et al. "A Hybrid Particle Swarm Optimization for Parallel Machine Scheduling with Shared and Multi-mode Resources". In: *IEEE Transactions on Computational Intelligence and Design* 9.4 (Dec. 2023), pp. 2473–2479. DOI: 10.1109/TGCN.2023.3283509.

[26] Bong Chia Lih, Sze San Nah, and Noor Alamshah Bolhassan. "A study on heuristic timetabling method for faculty course timetable problem". In: (2018).

[27] Yanming Yang, Wanchun Gao, and Yang Gao. "Mathematical modeling and system design of timetabling problem based on improved GA". In: (2017).

[28] Mingqiang Gao and Xu Yang. "APSO-SL: An Adaptive Particle Swarm Optimization with State-Based Learning Strategy". In: *Processes* 12 (2024), p. 400. DOI: 10.3390/pr12020400.

[29] Saeed Fallahi and Mohamadreza Taghadosi. "Quantum-behaved Particle Swarm Optimization Based on Solitons". In: *Scientific Reports* 12 (2022), p. 13977. DOI: 10.1038/s41598-022-18351-0.

[30] Tianyu Liu et al. "Quantum-behaved Particle Swarm Optimization with Collaborative Attractors for Nonlinear Numerical Problems". In: *Communications in Nonlinear Science and Numerical Simulation* 44 (2017), pp. 167–183. DOI: 10.1016/j.cnsns.2016.08.001.

[31] Zhi-Hui Zhan et al. "Adaptive Particle Swarm Optimization". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics* 39.6 (Dec. 2009), pp. 1362–1375. DOI: 10.1109/TSMCB.2009.2015956.

[32] Qi Liu et al. "A Multi-Swarm PSO Approach to Large-Scale Task Scheduling in a Sustainable Supply Chain Datacenter". In: *IEEE Transactions on Green Communications and Networking* 7.4 (Dec. 2023), pp. 1667–1678. DOI: 10.1109/TGCN.2023.3283509.

[33] Tim Blackwell, Jürgen Branke, and Xiaodong Li. "Particle Swarms for Dynamic Optimization Problems". In: (2008), pp. 193–217.

[34] K. E. Parsopoulos and M. N. Vrahatis. "Recent approaches to global optimization problems through Particle Swarm Optimization". In: *Natural Computing* 1.2-3 (2002), pp. 235–306. DOI: 10.1023/A:1016568309421.