Hybrid Genetic Algorithm - Particle Swarm Optimization (Hybrid GA-PSO)

1. Introduction

Due to the problem's NP-hard nature and the large, highly constrained search space, traditional deterministic approaches often fall short in finding feasible or high-quality solutions within a reasonable time frame. As a result, metaheuristic techniques have become a popular alternative.

This paper presents a hybrid optimization approach that combines Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) to solve the university course timetabling problem. The hybrid GA-PSO algorithm aims to balance **exploration**—the ability to broadly search the solution space—and **exploitation**—the ability to intensively improve promising solutions. The PSO phase performs global search to quickly locate high-quality regions in the search space, while the GA phase refines the best solutions using evolutionary operators such as crossover, mutation, and local search heuristics. This synergy between the two methods enhances both the convergence speed and the final solution quality, addressing the limitations of using each technique in isolation.

2. Hybrid GA-PSO Approach

The **Hybrid GA-PSO** framework consists of **two optimization phases**:

2.1 Phase 1: Particle Swarm Optimization (PSO)

1. **Swarm Initialization**:
   - Each particle represents a candidate schedule (set of courses assigned to rooms and timeslots).
   - Particles begin with **random assignments** and velocities.
2. **Fitness Evaluation**:
   - A fitness function evaluates **schedule feasibility**, penalizing conflicts.
3. **Velocity and Position Updates**:
   - Each particle adjusts based on **inertia**, its **personal best solution**, and the **global best solution**.
4. **Stopping Criteria**:
   - If there is no improvement for **multiple iterations**, an **early stop mechanism** terminates PSO.
5. **Top Solutions Selection**:
   - The **best-performing particles** are passed to the GA phase for further optimization.

2.2 Phase 2: Genetic Algorithm (GA)

1. **Population Initialization**:
   - The best solutions from PSO form the **initial GA population**.

2. **Selection Mechanism**:
   - GA applies **elitism**, ensuring the best individual survives.
   - Parents are selected from **top-performing solutions**.
3. **Crossover & Mutation**:
   - **One-point crossover** creates new individuals by combining parent schedules.
   - **Mutation** introduces diversity by modifying course assignments.
4. **Local Search Refinement**:
   - A heuristic-based **local search** improves individuals by testing **small mutations**.
5. **Stopping Criteria**:
   - If an optimal fitness score (no conflicts) is achieved, GA terminates early.

3. Algorithm parameters setup

PSO Swarm size	250 Particles
PSO Iterations Number	100 Iterations
Genetic Iterations Number	250 Iterations
Inertial weight	0.5
Cognitive Learning Rate	1.5
Social Learning Rate	2
Genetic Population Size	30 Chromosomes
Crossover Rate	0.7
Mutation Rate	0.01
Mutation Type	Single point Mutation
Crossover Type	One Point Crossover
Selection Type	Elitism + Tournament selection

3. Algorithm Pseudocode

Algorithm: Hybrid Genetic Algorithm - Particle Swarm Optimization (Hybrid GA-PSO)

Input:

- **PSO Parameters:**
  - Swarm size (**S**)
  - Number of iterations (**max_iterations**)
  - Inertia, cognitive, and social factors
- **GA Parameters:**
  - Population size (**P**)
  - Number of iterations (**ga_iterations**)
  - Crossover rate, mutation rate

Phase 1: Particle Swarm Optimization (PSO)

1. **Initialize a swarm of S particles**, each representing a random timetable.
2. Compute **fitness** for each particle and identify the **global best solution**.
3. **Iterate for max_iterations:**
   - For each particle:
     1. Evaluate fitness and update personal best if improved.
     2. Update global best solution if fitness is better than previous best.
     3. Adjust velocity using **inertia, cognitive influence, and social influence**.
     4. Update position based on velocity, ensuring valid constraints.
   - If no improvement occurs for several iterations, **trigger early stopping**.
4. **Select top-performing solutions** to serve as the initial population for GA.

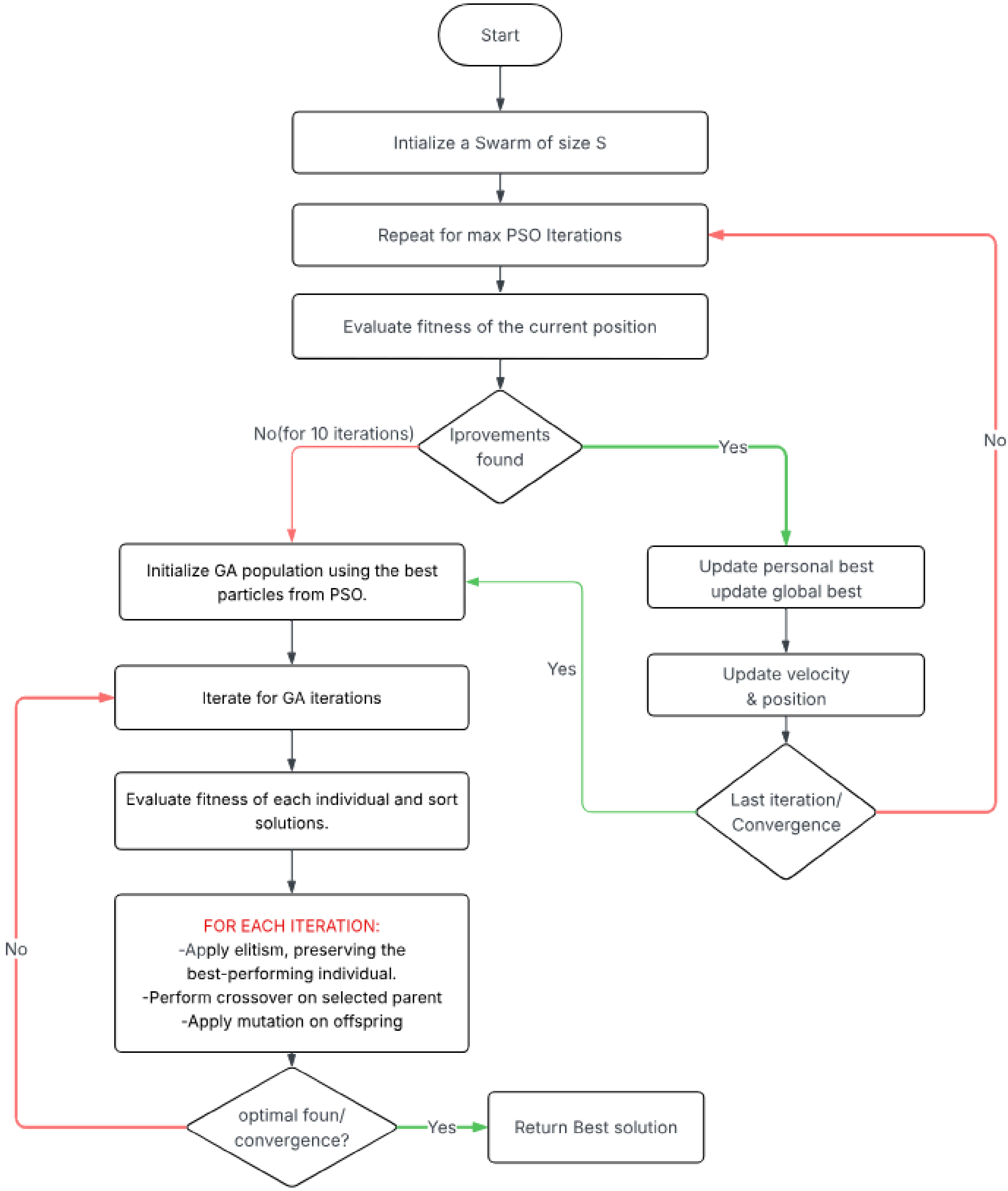
Phase 2: Genetic Algorithm (GA)

1. **Initialize GA population** using the best particles from PSO.
2. **Iterate for ga_iterations:**
   - **Evaluate fitness** of each individual and sort solutions.
   - Apply **elitism**, preserving the best-performing individual.
   - Perform **crossover** on selected parent pairs to generate new offspring.
   - Apply **mutation** on offspring with a probability.
   - Conduct **local search** to refine solutions further.
3. **Repeat until reaching convergence** or achieving an optimal fitness score.

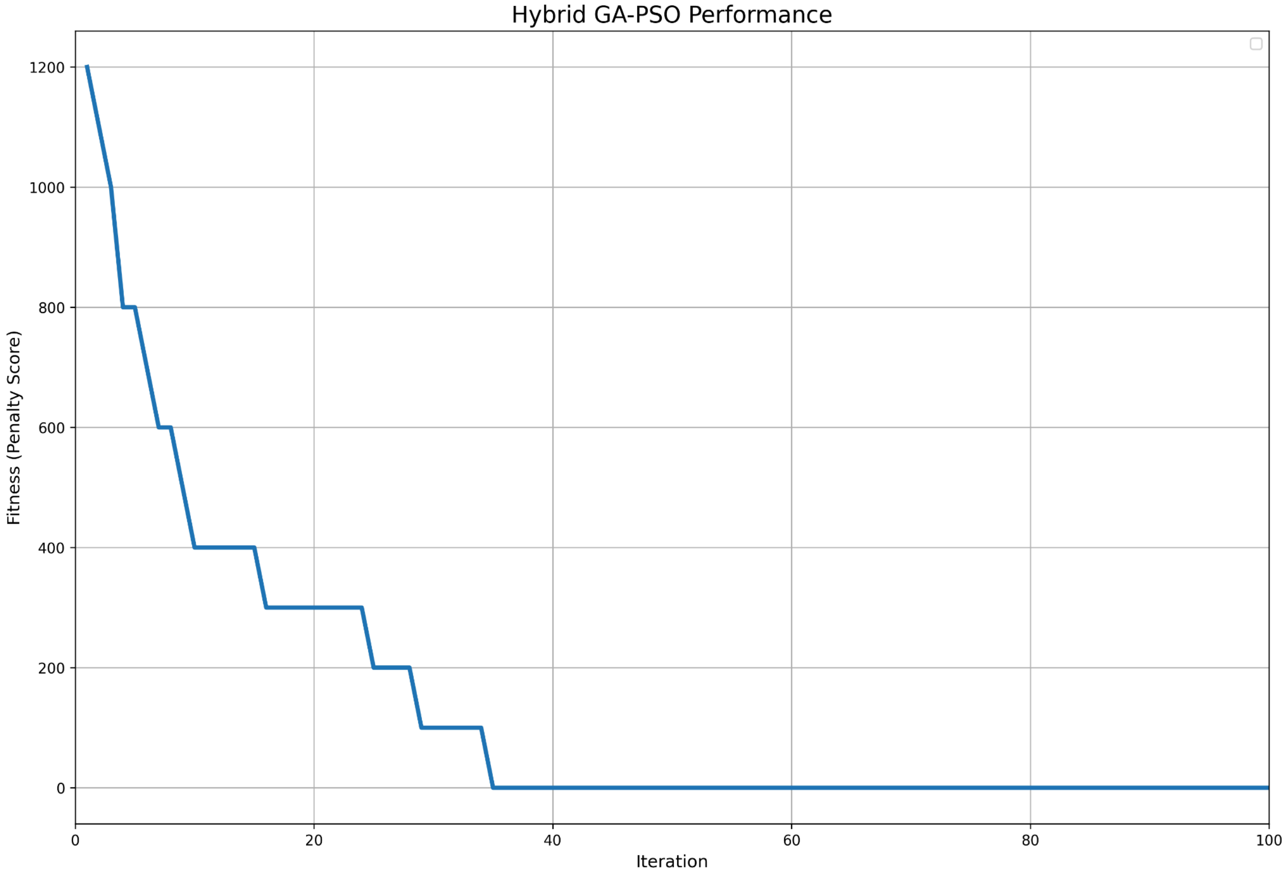
Output:

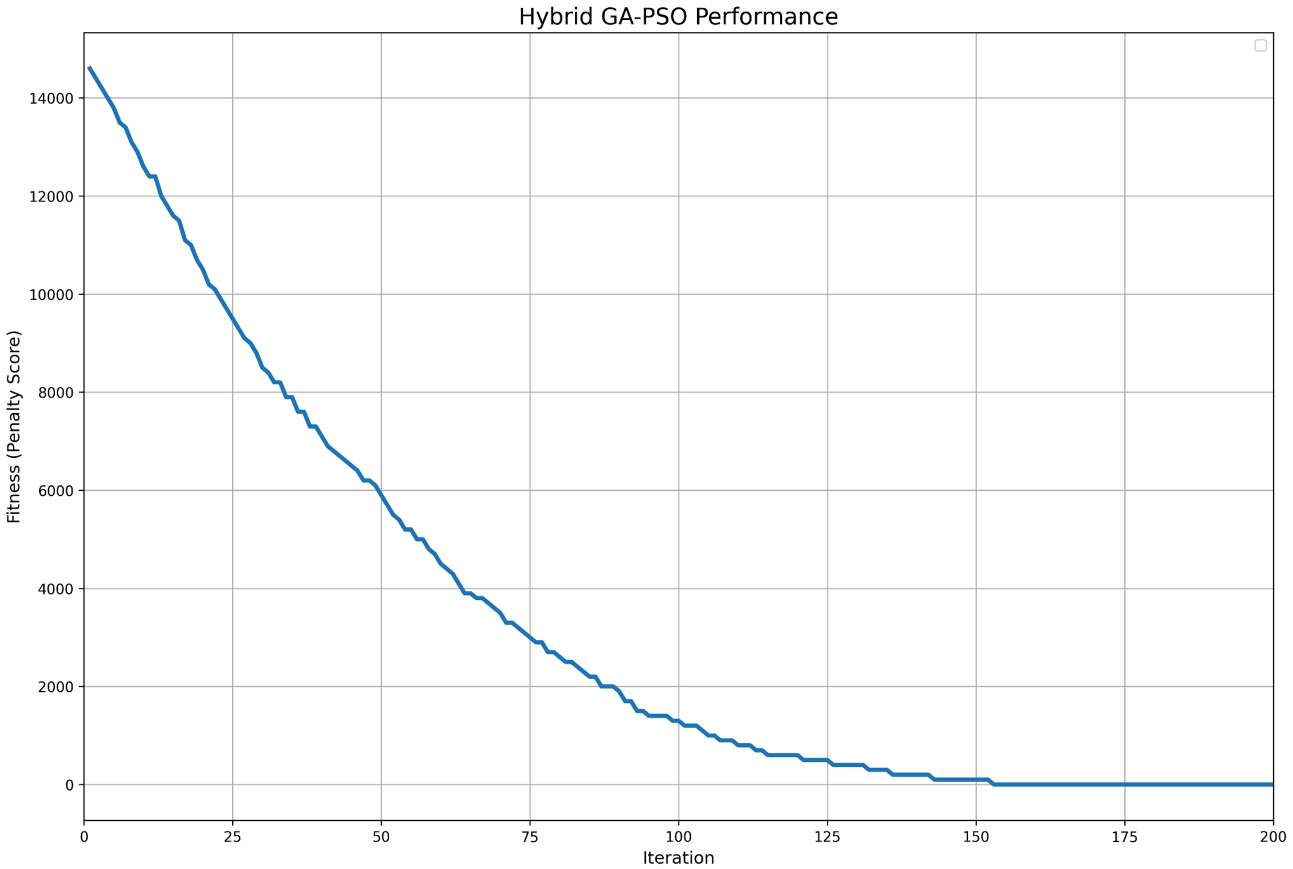
- The **best optimized timetable** with **minimum conflicts and improved scheduling efficiency**.

4. Algorithm Flowchart



5. Experimental Setup and results





6. Behavioral Analysis of Hybrid GA-PSO from Results

The **Hybrid Genetic Algorithm-Particle Swarm Optimization (Hybrid GA-PSO)** combines **PSO's global search** with **GA's local refinement** to enhance optimization efficiency. The results reveal **distinct behavioral patterns** across the two optimization phases.

Phase 1: PSO Optimization Behavior

- **Rapid Initial Fitness Improvement:**
    - Early iterations maintain **high fitness values** (above **20,000**) with minor fluctuations.
    - The first noticeable improvement appears at **Iteration 13**, with a drop from **21,300 to 20,000**, indicating particles start refining solutions.
- **Gradual Fitness Decline:**
    - The swarm continuously improves until around **Iteration 94**, reaching **14,600**.
    - **Early stopping at Iteration 95** suggests PSO stagnated and could no longer refine solutions effectively.
- **Limitation:**
    - PSO **quickly finds promising solutions**, but once particles **settle into local optima**, further improvement slows or stops.

Phase 2: Genetic Algorithm (GA) Optimization Behavior

- **Continuous Refinement Post-PSO:**
    - GA starts with **PSO's best solution (14,600 fitness)** and progressively **refines the solution** using crossover, mutation, and local search.
    - Fitness **steadily improves**, reaching **8,500 within 30 iterations**.
- **Sustained Exploration and Adaptability:**
    - Unlike PSO, GA **does not stagnate**.
    - It continues refining, **reducing the fitness score to 0**, meaning an **optimal solution was found**.
- **Final Convergence to Optimality:**
    - **Near-zero fitness values** in later iterations highlight GA's **strong local search capabilities**, compensating for PSO's stagnation.

Observations from Hybrid Behavior

1. **PSO rapidly finds near-optimal solutions**, making it effective in **global exploration**.
2. **GA refines the solutions**, ensuring continued improvement beyond PSO’s stagnation.
3. The hybrid approach **overcomes the weaknesses of standalone PSO or GA**, resulting in **efficient and reliable optimization**.

Thus, **Hybrid GA-PSO effectively balances exploration (PSO) and exploitation (GA)**, leading to robust **solution convergence** while avoiding premature stagnation.