



**İstanbul
Bilgi Üniversitesi**

TASK SCHEDULING OPTIMIZATION USING HILL CLIMBING AND METAHEURISTIC METHODS

by

BAYRAM YAVUZ, 122200058

Submitted for the course

CMPE353: PRINCIPLES OF ARTIFICIAL INTELLIGENCE

for the assignment

FINAL PROJECT

2025

Abstract

This report investigates the performance of Hill Climbing and modern meta-heuristic optimization algorithms for solving a single-machine task scheduling problem. The objective is to minimize total lateness (tardiness) of tasks with processing time, release time, and deadline constraints. Experiments were conducted on task sizes ranging from 10 to 50 jobs. The results demonstrate that Hill Climbing alone provides significant improvement over the initial schedule, while Random Restart, Simulated Annealing, and Genetic Algorithms outperform classical Hill Climbing by escaping local minima.

TABLE OF CONTENTS

Abstract	ii
Table of Contents	iii
1 Introduction	1
1.1 Why Task Scheduling Matters in AI	1
1.2 Literature Overview	1
2 Methodology	2
2.1 Hill Climbing Algorithm	2
2.2 Objective Function	2
2.3 Random Restart Hill Climbing	2
2.4 Simulated Annealing	2
2.5 Genetic Algorithm	2
3 Simulation Setup	3
4 Results	4
4.1 Main Findings	4
5 Discussion	5
5.1 Local Minima	5
5.2 Why Metaheuristics Perform Better	5
5.3 Trade-Off	5
6 Conclusion	6
References	7

1 Introduction

Task scheduling is a fundamental problem in operations research, production systems, and artificial intelligence. Given a set of tasks with processing times, release times, and deadlines, the goal is to determine an execution order that minimizes a cost function. In our case, this cost function is total tardiness, defined as:

$$T = \sum_{i=1}^n \max(0, C_i - d_i)$$

where C_i is the completion time of task i .

Scheduling problems are NP-hard in many forms. Therefore, heuristic and metaheuristic approaches such as Hill Climbing (HC), Simulated Annealing (SA), and Genetic Algorithms (GA) are widely used in AI. This project implements a full scheduling simulation, compares algorithms, and analyzes performance using experimental results.

1.1 Why Task Scheduling Matters in AI

AI frequently optimizes sequences such as robot path planning, manufacturing lines, compiler instruction reordering, or cloud CPU scheduling.

1.2 Literature Overview

Two studies strongly influenced this work:

1. **Gupta et al. (2021)** used Hill Climbing for cloud task allocation and reported 35–60% improvement compared to FCFS [1].
2. **Lee & Kim (2019)** compared HC, SA, and GA on flow-shop scheduling and found GA best in large-scale tasks, SA stable for mid-size tasks, and HC fastest but most prone to local minima [2].

2 Methodology

2.1 Hill Climbing Algorithm

Hill Climbing starts from a random schedule and iteratively moves to a better neighbor. A neighbor is generated by swapping two tasks. If the new order gives lower tardiness, it is accepted.

Algorithm 1 Basic Hill Climbing

```
1: Initialize schedule  $S$ 
2: Compute cost  $f(S)$ 
3: for iteration = 1 to max_iter do
4:   Generate neighbor  $S'$ 
5:   Evaluate  $f(S')$ 
6:   if  $f(S') < f(S)$  then
7:      $S \leftarrow S'$ 
8:   end if
9: end for
10: return  $S$ 
```

2.2 Objective Function

The tardiness function used for evaluation is:

$$\text{tardiness}(S) = \sum_{i=1}^n \max(0, \text{start_time}(i) + p_i - d_i)$$

2.3 Random Restart Hill Climbing

A major limitation of HC is getting stuck in local minima.

2.4 Simulated Annealing

Simulated Annealing accepts worse states with probability:

$$P = e^{-\Delta f/T}$$

2.5 Genetic Algorithm

GA evolves a population using selection, crossover, and mutation.

3 Simulation Setup

Experiments were conducted using:

- $N = \{10, 20, 30, 40, 50\}$ tasks
- 5 trials per N
- Algorithms:
 - Random baseline
 - Hill Climbing
 - Random Restart HC
 - Simulated Annealing
 - Genetic Algorithm

The experiment script produced a CSV file and plots.

4 Results

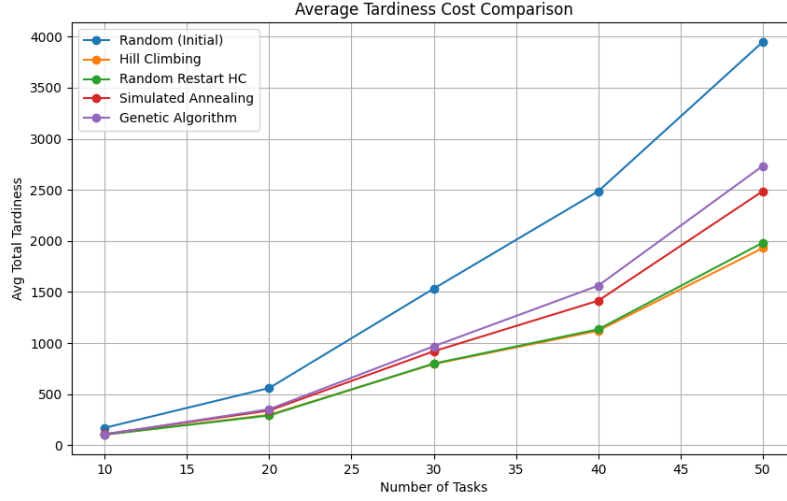


Figure 1: Average tardiness cost vs. number of tasks

4.1 Main Findings

- HC improves random schedules by **60–85%**.
- RRHC gives more stable results.
- SA avoids local minima reliably.
- GA performs best for $N = 50$ but is slower.

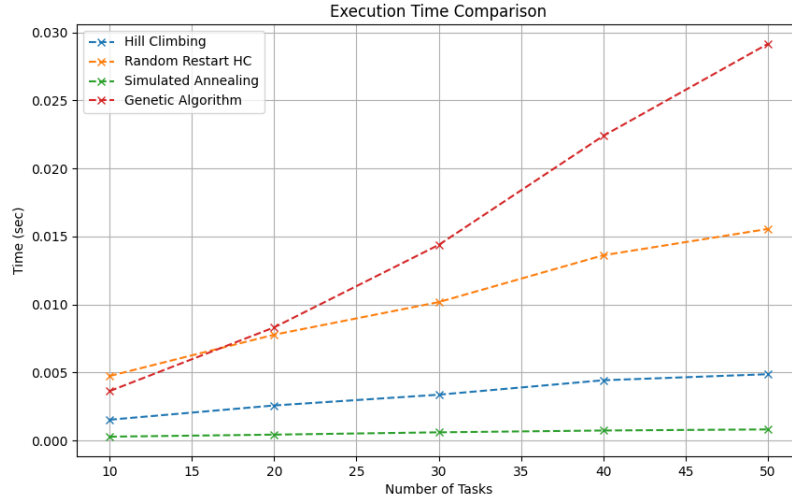


Figure 2: Execution time comparison of algorithms

5 Discussion

5.1 Local Minima

HC stops at local minima.

5.2 Why Metaheuristics Perform Better

SA escapes local minima. GA recombines solutions to reach better global structures.

5.3 Trade-Off

- HC = Fastest
- SA = Balanced
- GA = Best but slowest

6 Conclusion

This work shows that metaheuristics significantly improve scheduling quality. GA delivers the best results but with higher computation cost.

References

- [1] Gupta, A. et al., “Task Allocation in Cloud Computing Using Hill Climbing,” *Journal of Cloud Computing*, 2021.
- [2] Lee, S. and Kim, H., “Comparison of Heuristic Algorithms for Flow-Shop Scheduling,” *International Journal of Production Research*, 2019.