Exercise 1 Analysis

1. Introduction

This report compares the performance of three optimization algorithms; Random Local Search (RLS); the (1+1) Evolutionary Algorithm (EA11); and a Genetic Algorithm (GA); on a set of graph-based combinatorial problems. The aim is to understand how algorithmic complexity and limited evaluation budgets influence performance across problems with different structures.

2. Experimental Setup

Each algorithm was tested on eleven problem instances that belong to three main classes; MaxCoverage (2100–2103); MaxInfluence (2200–2203); and PackWhileTravel (2300–2302). Every algorithm–problem combination was repeated thirty times using a fixed budget of 10,000 fitness evaluations. Performance was measured through mean fitness trajectories with standard deviation confidence bands.

RLS performs single-bit flips with greedy acceptance; EA11 applies bit-flip mutation with elitist selection; and GA uses tournament selection, uniform crossover, and bit-flip mutation with a population of 50.

3. Results

MaxCoverage Problems:
RLS improved rapidly in the early stages and usually plateaued by around 2,000 evaluations. For instances 2100 and 2101, all algorithms reached similar final fitness (around 420); however, RLS achieved this level much faster than EA11 or GA, which required up to 6,000 evaluations. In 2102, EA11 and GA slightly outperformed RLS (520–530 vs 480). Instance 2103 was the most difficult, showing greater variance (±50–100) and a final fitness near 600. These patterns reflect the greedy behaviour of RLS in exploiting submodular gains before reaching diminishing returns.

MaxInfluence Problems:
This class displayed stronger differences in performance. All algorithms produced negative fitness values (–2000 to –200/–1300), showing constraint violations. RLS consistently achieved the best results (–200 to –450), EA11 was moderate (–450 to –600), and GA performed worst (–1100 to –1300). The near-linear improvement suggests the search struggled to find feasible solutions, though the results were stable with low variation between runs.

PackWhileTravel Problems:
Instances 2300 and 2301 were relatively easy; all algorithms converged quickly within 500–2,000 evaluations to fitness values between 25,000 and 400,000. The third instance, 2302, was far more challenging, showing clear separation; RLS reached 1.3 million; EA11 1.29 million; and GA 1.1 million. This ranking is consistent with the trend seen in MaxInfluence.

4. Discussion

Across almost all problems, RLS performed as well as or better than the more complex algorithms. Its greedy hill-climbing approach fits well with submodular problem structures, where improvements yield diminishing returns. The restricted 10,000-evaluation budget placed population-based methods at a disadvantage; GA's population size of 50 limited it to roughly 200 generations, reducing its ability to balance exploration and exploitation. RLS, by focusing all evaluations on one trajectory, used the budget more efficiently. GA's weaker performance also indicates that crossover may disrupt useful relationships in graph-based representations; EA11's mutation-only strategy avoided this issue and produced more consistent results. Variance trends further support these findings; RLS showed stable convergence; EA11 moderate variation; and GA the most inconsistency during mid-search.

5. Conclusion

The analysis demonstrates that algorithm performance on graph-based submodular problems depends heavily on problem structure, search mechanisms, and evaluation limits. RLS proved highly effective due to its simplicity and efficient use of the budget; EA11 provided balanced and reliable results; while GA's complexity and reliance on crossover limited its performance under tight constraints. These findings suggest that simpler local search methods can outperform population-based algorithms when resources are limited. Future work could explore whether larger budgets allow GA to exploit population diversity more effectively or whether graph-aware crossover strategies could enhance its outcomes.