Université libre de Bruxelles
École Polytechnique de Bruxelles
M-IRIFS

INFO-H413 - Heuristics optimization - 2023/2024

Linear ordering problem

Implementation exercise 2

Student:
Ismail Jeq

Teacher:
Thomas STÜTZLE

May 10, 2024

Contents

Abstract

In this study, we investigate stochastic local search (SLS) algorithms implemented in C++ for solving the linear ordering problem (LOP) across various instances. Our approach includes iterated local search (ILS) with with an initial solution based on Chenery and Watanabe, insert-based local search, exchange-based perturbation, and simulated annealing (SA) with an initial solution based on Chenery and Watanabe, along with insert local search. We evaluate the performance of each algorithm on instances of size 150, using metrics such as average percentage deviation from the best-known solutions and total computation time. Additionally, we employ statistical tests (Student t-test and Wilcoxon test) to assess whether there are statistically significant differences between the solutions generated by each algorithm.

Keywords: linear ordering problem, stochastic local search, simulating annealing, iterated local search , Student t-test, Wilcoxon test.

1 Introduction

The objective of this second practical task was to devise and execute two stochastic local search (SLS) algorithms from two distinct categories (simple, hybrid, or genetic) with the aim of resolving the linear ordering problem. In this research, we opted to implement ILS (a hybrid algorithm) and SA (a simple algorithm). A collection of instances with dimensions of 150 and 250, along with their respective optimal solutions, were provided for this task and used as benchmarks for the algorithms we developed.

The termination criteria for the algorithms were determined by the average computational time required to execute a complete VND, which was implemented in the initial task, on the same instance, and then multiplied by 100 to ensure sufficient runtime for the SLS algorithms. In order to assess the effectiveness of each algorithm, we calculated statistics such as average percentage deviation and computational time for all instances. We also conducted statistical analyses using the Student-t test and the Wilcoxon test to ascertain if there were significant disparities between the solutions. Additionally, we generated correlation plots between the two algorithms for all computed instances.

2 Problem description

2.1 Linear ordering problem (LOP)

The Linear Ordering Problem (LOP) is a type of combinatorial optimization problem that focuses on organizing a set of items in a specific linear order to optimize a given objective function. The complexity of this problem stems from the need to find an optimal permutation of elements from a defined set that satisfies certain criteria.

In mathematical terms, the LOP can be described in this way: We are given a matrix C, of dimensions $n \times n$. The goal is to discover a permutation π, which is a rearrangement of the indices ranging from 1 to n, that results in the maximization of the sum of elements located in the upper triangle of the matrix C. The representation of this objective function is as follows:

$$f(\pi) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} c_{\pi_i \pi_j} \tag{1}$$

where $c_{\pi_i \pi_j}$ denotes the element of the matrix C located at row π_i and column π_j after the permutation π has been applied. [1]

The Linear Ordering Problem holds substantial relevance in numerous fields of study. In the realm of economics, it plays a pivotal role in the triangularization of input-output matrices, thereby promoting effective distribution of resources. Within the scope of sociology, it proves instrumental in deciphering social hierarchies and preferences.

Furthermore, in the domain of graph theory, linear orderings are indispensable for the representation of partial orders and task scheduling. In addition, in the field of archaeology, this problem is utilized to determine chronological sequences, as demonstrated by the application of the Harris Matrix. [2]

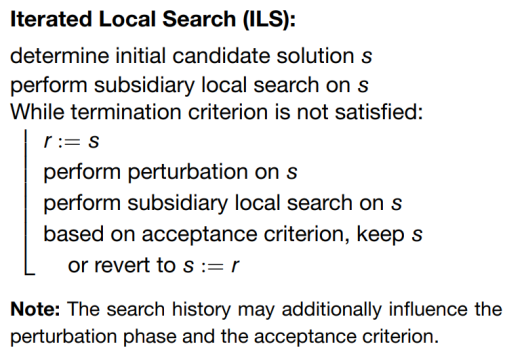
3 Material and methods

3.1 Data

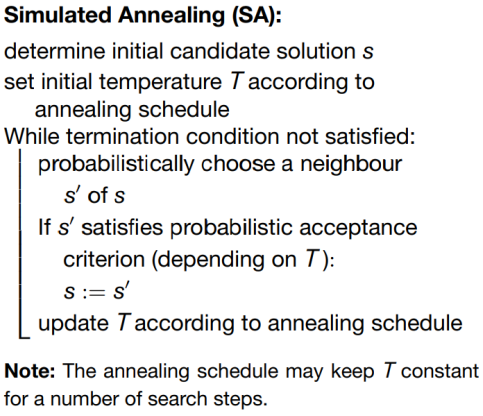
The data used in this study consists of instances provided by the teacher. These instances vary in size between 150 and 250. For this second implementation exercise, only instances of size 150 are used. These can be found in the folder `./instances/`.

3.2 SLS algorithms

For this second implementation exercise, we implemented two stochastic local search (SLS) algorithm from two different classes (simple, hybrid or genetic) in order to solve the linear ordering problem. We selected ILS (hybrid) and SA (simple) [1]:



(a) Iterated local search



(b) Simulated annealing

Figure 1: SLS algorithms

3.2.1 Iterated local search (ILS)

The Iterated Local Search (ILS) algorithm is a powerful metaheuristic optimization technique that iteratively refines a candidate solution by combining local search and perturbation phases. In our implementation, we focus on the following key components:

- **Initialization**:We start by generating an initial solution using the Chenery-Watanabe approach. Specifically, we utilize the matrix of technical coefficients to determine the interdependence among economic activities. The seed for this initialization is calculated as the sum of elements in the problem instances. This ensures consistent initial solutions across multiple runs.
- **Local Search**: Our local search procedure employs a first improvement insert-based strategy. It explores the largest neighborhood (compared to exchange and transpose) possible by visiting all j elements for a chosen i. The goal is to return the first solution improvement found in the neighborhood.
- **Perturbation**: To introduce diversification, we use a random exchange strategy to explore the neighborhood. This perturbation strategy cannot be reversed with insert movements in a single step, as mentioned in the reference[1]. It complements our local search. The number of exchange moves used in perturbation is a parameter in our algorithm. We choose the first improved solution to ensure that our perturbed solution consistently improves.
- **Acceptance Criterion**: The acceptance criterion assesses the perturbed solution by comparing its objective function value to that of the current solution. If the perturbed solution yields an improvement, it is retained for further exploration. Otherwise, we revert to the last best solution.
- **Termination**: The algorithm terminates if the elapsed time is superior to the max runtime computed in Section 4.1.1.

3.2.2 Simulated Annealing (SA)

The implemented Simulated Annealing (SA) is a simple local search algorithm that combines probabilistic technique with local search methods to efficiently explore the search space. The components and justifications for the SA algorithm are as follows:

- **Initialization**: Similar to ILS, the SA algorithm initializes an solution using the Chenery-Watanabe approach.
- **Neighbor Selection**: The neighborhood selection in the SA algorithm follows the same first improvement strategy as in ILS to explore the largest neighbourhood. It is also a first improvement rule insert-based strategy.
- **Temperature Schedule**: The temperature schedule in the SA algorithm employs a geometric cooling schedule, as it has been shown to produce the best results. The initial temperature is set to 4000.0 and the cooling rate (alpha) is 0.99.
- **Reheating**: The SA algorithm includes a reheating mechanism that increases the temperature by a factor of 1.2 every 100 iterations to escape local minima.
- **Acceptance Probability**: The acceptance probability function in the SA algorithm is based on the Metropolis criterion. If the new solution is worse than the current solution, it is accepted with a probability of 1 / (1 + exp(delta_score / T)), where delta_score is the difference in scores between the new solution and the current solution.

  At a high temperature, the algorithm is more likely to accept worse solutions, which allows it to explore the solution space more broadly and avoid getting trapped in local minima. This is the exploration phase.

As the temperature decreases, the algorithm becomes less likely to accept worse solutions and starts to fine-tune the current best solution. This is the exploitation phase.

- **Termination**: Similar to ILS, the SA algorithm terminates if the elapsed time is superior to the max runtime.

The selection of parameters and operators in both algorithms is influenced by empirical testing, findings from prior research, and the specific characteristics of the problem at hand. This approach ensures that the algorithms perform effectively and consistently converge towards high-quality solutions.

4 Results

All these results can be obtained using the bash script (`./scripts/statistcics.sh`) or by running `make satistics` with Makefile. The results are accessible in the `./statistics/experiments/` folder.

4.1 Statistics

4.1.1 Termination criterion

The termination criterion for the Stochastic Local Search (SLS) algorithms is determined by the mean computational time required to execute a complete Variable Neighborhood Descent (VND), as implemented in the initial exercise. This time is then amplified by a factor of 100 to ensure sufficient runtime for the SLS algorithms. For all instances with a size of 150, the average runtime is approximately 600 seconds. The details of these runtimes can be located in `./max_run_time/max_runtime.txt`.

4.1.2 Average percentage deviations and total computation time

For each algorithm, the average deviation (in %) and total computation time (in second) is calculated on instances of size 150. The Table 1 summarise these informations for ILS and MA. Both ILS and SA give similar results:

Configuration	**Average deviation**	**Total computation time**
ils_cw_insert_first	2.025635	22768.213000
sa_cw_insert_first	2.043718	22765.364000

Table 1: Average deviation (in %) and total computation time (in second) for each configuration . Note that the total computation time is different because some iterations are longer for different algorithms.

4.2 Statistical tests

These statistical evaluations employ a proposition known as the null hypothesis. In this context, it states: "the median difference between the outcomes of two experiments is zero". We set the significance level, denoted as α , to 0.05. If the calculated p-value falls below α , we reject the null hypothesis. In this scenario, it implies that there is a noteworthy distinction between the experiments. An R script, named as `./statistical_tests.r`, was crafted to implement these tests on every conceivable pair of experiments. The results were then stored in the. The output was saved in the `./statistics/statistical_tests/`.

4.2.1 Student-t test and Wilcoxon test

The results of the statistical tests shown in Table 2 do not indicate significant differences between the ILS and SA algorithms. The Student-t test yielded an relatively important p-value (0.747), providing strong evidence for the null hypothesis and thus not suggesting a significant difference in performance between the two algorithms. Similarly, the Wilcoxon test also produced a relarively important p-value (0.746), further supporting the acceptance of the null hypothesis and indicating a statistically similarity in performance between the ILS and SA algorithms.

Algorithm	ILS	MA
ILS	0	0.747
MA	0	0

(a) Student-t

Algorithm	ILS	MA
ILS	0	0.746
MA	0	0

(b) Wilcoxon

Table 2: Statistical tests

4.3 Correlation plot

To evaluate the relationship between the performance of Iterated Local Search (ILS) and Simulated Annealing (SA), we constructed a scatter plot that depicts the association between their respective relative deviations. Furthermore, we computed the correlation coefficient, which came out to be exactly 0.5565183. This positive value of the correlation coefficient signifies a moderate positive linear correlation between the relative deviations of ILS and SA. Essentially, this means that an increase or decrease in the relative deviation for ILS is likely to be accompanied by a similar trend in the relative deviation for SA, albeit not in a perfect one-to-one manner.

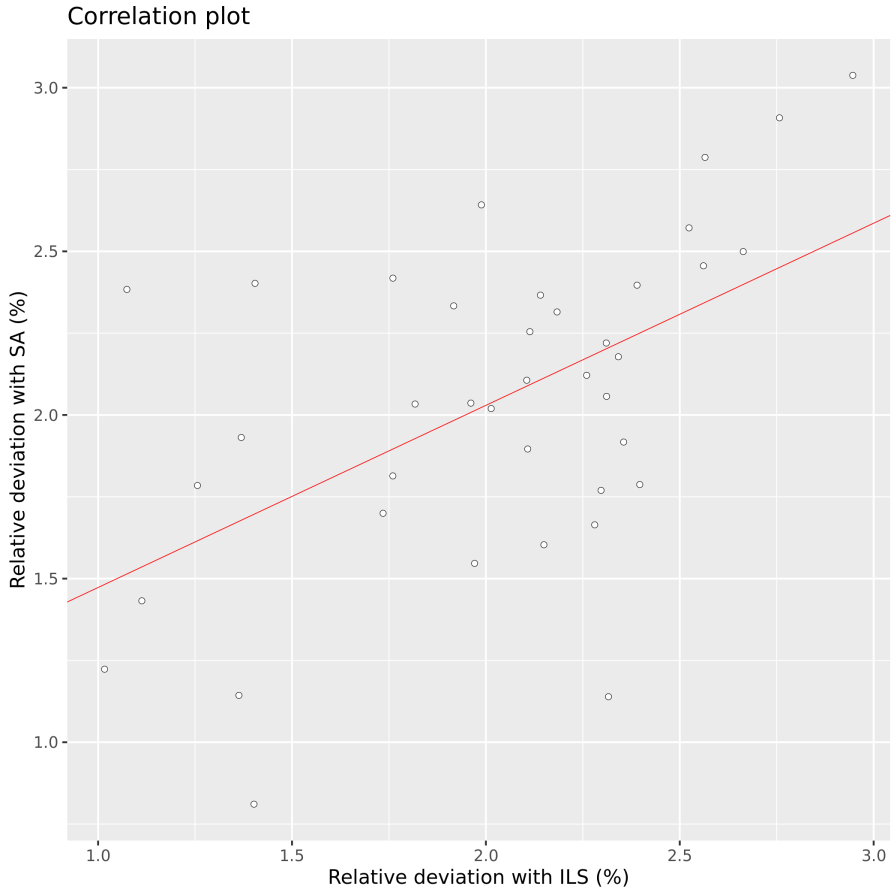


Figure 2: Correlation plot

5 Conclusion

A performance assessment was carried out on 39 instances, each of size 150, to gauge the effectiveness of two Stochastic Local Search (SLS) algorithms - Iterated Local Search (ILS) and Simulated Annealing (SA), in solving the Linear Ordering Problem (LOP). The most superior outcomes were produced by the ILS, with SA trailing just behind in close proximity.

References

[1] *The Linear Ordering Problem: Instances, Search Space Analysis and Algorithms*, Tommaso Schiavinotto & Thomas Stützle

[2] *Stochastic Local Search Foundations and Applications*, Elsevier, 2004, Holger H. Hoos & Thomas Stützle