Université Libre de Bruxelles

Heuristic optimisation INFO-H413

Implementation Exercise Report Linear Ordering Problem (LOP)

Student :
Achten Alexandre 494484

Professor :
Stützle Thomas

April 2024

Academic year 2023-2024

Abstract

"The LOP is $\mathcal{NP}$-hard, that is, we cannot expect to find a polynomial time algorithm for its solution" [SS04].
In this report, different algorithms will be exposed. Their mode of operation will be explained and the associated results (Section 3) will be presented. A statistical analyse (Subsection 3.3) between these results will be done using the Paired t-test and the Wilcoxon signed-rank test.
For the classical methods (Best and First) the initial solution have a significant impact on the most of the algorithms, at least when it's generated by the Cheneby and Watanabe heuristic.
Concerning the VND's algorithms, the order of evaluation of the neighbourhoods does have an impact.
The best-time algorithm is the transpose pivoting rule algorithm, but as it's relative deviation is really high, the Best exchange CW is the fastest one.
The most accurate algorithm, that have the lowest average relative deviation, is the First insert CW. But the statistical test showed that there is no significant difference with the Best insert CW, as the latter is faster, the Best insert CW is the best algorithm.

Contents

1 Introduction

The linear ordering problem (LOP) is a classical problem that arises in several different fields (economics, archeology,...). Given a matrix C nxn and a permutation π of the columns and indices $\{1, ..., n\}$, the sum of the elements in the upper right triangle must be maximised. More formally, this formula must be maximised:

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

"The LOP is $\mathcal{NP}$-hard, that is, we cannot expect to find a polynomial time algorithm for its solution" [SS04].
In this report, different algorithms will be exposed. Their mode of operation will be explained and the associated results (Section 3) will be presented. A statistical analyse (Subsection 3.3) between these results will be done using the Paired t-test and the Wilcoxon signed-rank test.

2 Iterative Improvement methods

"The $\mathcal{N}_X(\pi)$ (the neighborhood of a permutation π), is defined to be the set of all permutations that can be obtained by applying one interchange move to π" [SS04]. Where the interchange moves are the pivoting rules we are discussing in Subsection 2.3.
Iterative improvement methods mean that the algorithm tries to find a better solution within the neighbourhood, and from the new solution, searches through the neighbourhood again.
The general operation of the iterative improvement method is explained here :

π := GenerateInitialSolution()
if π is not a local optimum do
    pick a neighbour π' $\in \mathcal{N}$ such that F(π') > F(π)
    π := π'

where F(π) is the function to maximise.
The two main algorithms used here are first-improvement and best-improvement. Each has its advantages and disadvantages.

2.1 First-improvement

The First-Improvement method is an iterative improvement method,
This method rely on the principle that when evaluating neighbours, whenever it finds a neighbour that's better, it picks this solution without all other possibilities. It's initial solution can be generated randomly or using the Chenery and Watanabe (CW) heuristic construction method (Subsection 2.4).
The permutation can be modified in several ways, in this report 3 pivoting rules are used and are presented in the Subsection 2.3. Each pivoting rule has it's own neighbourhood.

2.2 Best-improvement

This method is based on the principle of exhaustive search of all neighbours. It remembers the best neighbour of all and returns it when it has finished evaluating each neighbour.
It's initial solution can be generated randomly or using the Chenery and Watanabe (CW) heuristic construction method (Subsection 2.4).
The permutation can be modified in several ways, in this report 3 pivoting rules are used and are presented in the Subsection 2.3. Each pivoting rule has it's own neighbourhood.

2.3 Pivoting rules

Each pivoting rule have it's own neighbourhood, so it's a very important choice to choose the kind of pivoting rule. These pivoting rules can be applied on any vector.

2.3.1 Exchange

The exchange pivoting rule consist in swapping any element of the vector with any other element. Let i be the elements to swap, the neighbourhood $\mathcal{N}_i$ are all $j \neq i$ elements such that $\forall j \in \pi$. An example is given in the Figure 1.

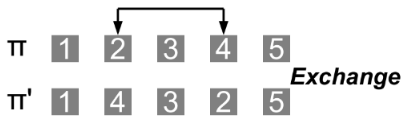

Figure 1: Exchange example

2.3.2 Transpose

The transpose pivoting rule consist in swapping an element with one of it's neighbours. Let i be the element to transpose, the neighbourhood $\mathcal{N}_i$ are all $j \neq i$ elements such that $\forall j \in \{i-1, i+1\}$. An example is given in the Figure 2.

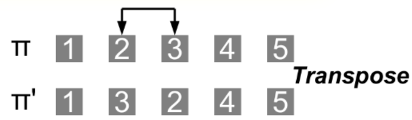

Figure 2: Transpose example

2.3.3 Insert

The insert pivoting rule insert the first element at the place of the second one, shifting the entire vector. An example is given in the Figure 3.

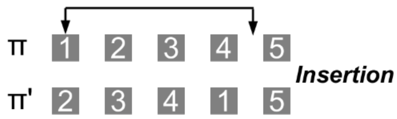

Figure 3: Insert example

2.4 Initial solution

At the beginning of any algorithm that searches for a solution, an initial solution is needed.

2.4.1 Random Solution

The random initial solution consists in creating a random solution that's acceptable. Concretely it generates a random vector that represents the permutation π.

2.4.2 Chenery and Watanabe

The Chenery and Watanabe is a constructive heuristic that creates the initial solution based on the attractiveness of each row, placing the most attractive one first. The attractiveness a_i of the row i is defined as the sum of the costs of each element in this row.

$$a_i = \sum_{j=1}^{n} c_{ij} \tag{2}$$

[MRD09]
Where c_{ij} is the cost of the element at row i and column j.

2.5 Variable Neighbourhood Descent

The Variable Neighbourhood Descent (VND) method is obtained when the change between neighbours is performed in a deterministic way. [MR11].
It follows the following schema :

k neighbourhoods $\mathcal{N}_1, ..., \mathcal{N}_k$
π := GenerateInitialSolution()
i:=1
repeat
    choose the first improving neighbour π' $\in \mathcal{N}_i(\pi)$
    if not $\exists\pi$' then i:=i+1
    else π := π' and i:=1
until $i > k$

In other words, if no other better neighbour is found, the algorithm searches through another neighbourhood by changing the pivoting rule.

In this report, two different descent are used :

- VND1 : Transpose $\rightarrow$ Exchange $\rightarrow$ Insert

- VND2 : Transpose → Insert → Exchange

In addition of these two descent, the two algorithms (VND1 and VND2) use the first improvement method and the Chenery and Watanabe initial solution.

3 Results

All the results are obtained on the XLOLIB dataset with $n = 150$ and $n = 250$ instances. The relative deviation compared to the best known solution is denoted Δ and the time to finish the algorithm is t. The tables below show the average relative deviation Δ_{avg} and the average time t_{avg}.

3.1 Iterative improvement

The first improvement algorithms were very slow, so it was not possible to test it on the $n = 250$ instances. The results are presented in the Table 1.
The names of the algorithms use this format :
"name_of_improvement" "name_of_pivoting_rule" "initial_solution".

Algorithm	Δ_{avg} (%)	t_{avg} (s)
First exchange random	2.700	30.218
First insert random	1.972	106.644
First transpose random	35.992	0.055
First exchange CW	2.469	28.010
First insert CW	1.658	73.659
First transpose CW	20.020	0.053

Table 1: First improvement algorithms on instances of $n = 150$

For the best improvement algorithms, instances of both sizes ($n = 150$ and $n = 250$) worked in reasonable time. The results for $n = 150$ are presented in Table 2 and for $n = 250$ at Table 3.

Algorithm	Δ_{avg} (%)	t_{avg} (s)
Best exchange random	3.714	0.756
Best insert random	2.091	1.587
Best transpose random	35.992	0.008
Best exchange CW	3.555	0.695
Best insert CW	1.989	1.421
Best transpose CW	20.020	0.006

Table 2: Best improvement algorithms on instances of $n = 150$

Algorithm	Δ_{avg} (%)	t_{avg} (s)
Best exchange random	3.543	6.517
Best insert random	2.228	15.228
Best transpose random	33.220	0.035
Best exchange CW	3.328	6.392
Best insert CW	1.936	15.106
Best transpose CW	18.667	0.036

Table 3: Best improvement algorithms on instances of $n = 250$

3.2 VND

Since the VND1 and VND2 use the same first improvement method, only $n = 150$ results can be provided. The VND algorithm uses CW initial solutions. The results are shown in the Table 4.

Algorithm	Δ_{avg} (%)	t_{avg} (s)
VND1	1.889	21.867
VND2	1.691	34.107

Table 4: VND algorithms on instances of $n = 150$

3.3 Statistical tests

In order to determine whether there is a statistical difference between the results of the different algorithms, 2 tests were carried out between them: the paired t-test (or Student's t-test) and the Wilcoxon signed rank test. The significance level α is put to $\alpha = 0.05$. The hypothesis that the distributions have the same average is rejected when p-value $< \alpha$.
The test were made over the relative deviation from the best known solution. The algorithm 1 is compared to the algorithm 2.

The results of the statistical tests on the different initial solution used (random and CW) are shown in the Table 5. We can conclude that the initial solution generated by CW heuristic have a significant impact on the Best exchange, Best insert and First exchange.

Algorithm 1	Algorithm 2	T-test p-value	Wilcox p-value
Best exchange random	Best exchange CW	0.2161	0.336
Best insert random	Best insert CW	0.297	0.468
Best transpose random	Best transpose CW	0	3.638e-12
First exchange random	First exchange CW	0.04969	0.09014
First insert random	First insert CW	0.002403	0.00155
First transpose random	First transpose CW	0	3.638e-12

Table 5: Statistical test on random and CW algorithms

The results of the statistical tests on the different iterative improvements (Best and First) are shown in the Table 6. We can conclude that the only algorithm where it makes a statistically difference is the insert random algorithm.

Algorithm 1	Algorithm 2	T-test p-value	Wilcox p-value
Best exchange random	First exchange random	1.366e-10	5.093e-11
Best exchange CW	First exchange CW	1.616e-14	3.638e-12
Best transpose random	First transpose random	0	0
Best transpose CW	First transpose CW	0	0
Best insert random	First insert random	0.247	0.336
Best insert CW	First insert CW	5.197e-06	4.403e-06

Table 6: Statistical test on Best and First algorithms

The results of the statistical tests on the VND algorithms are shown in the Table 7. We can conclude from the Wilcoxon test that there is a significant statistical difference between the VND1 and VND2. Thus the order of evaluation of the different neighbourhood have an impact on the performance of the algorithm.

Algorithm 1	Algorithm 2	T-test p-value	Wilcox p-value
VND1	VND2	0.03832	0.05435

Table 7: Statistical test on VND algorithms

4 Conclusion

To finish this report, we can summarize what have been illustrated in the Section 3 Results.
For the classical methods (Best and First) the initial solution have a significant impact on the most of the algorithms, at least when it's generated by the Cheneby and Watanabe heuristic.
Concerning the VND's algorithms, the order of evaluation of the neighbourhoods does have an impact.
The best-time algorithm is the transpose pivoting rule algorithm, but as it's relative deviation is really high, the Best exchange CW is the fastest one.
The most accurate algorithm, that have the lowest average relative deviation, is the First insert CW. But the statistical test showed that there is no significant difference with the Best insert CW, as the latter is faster, the Best insert CW is the best algorithm.

References

[MR11] Rafael Martí and Gerhard Reinelt. *The Linear Ordering Problem : Exact and heuristics methods*. en. 2011.

[MRD09] Rafael Martí, Gerhard Reinelt, and Abraham Duarte. *A benchmark library and a comparison of heuristic methods for the linear ordering problem*. en. 2009.

[SS04] Tommaso Schiavinotto and Thomas Stützle. *The Linear Ordering Problem: Instances, Search Space Analysis and Algorithms*. en. 2004.