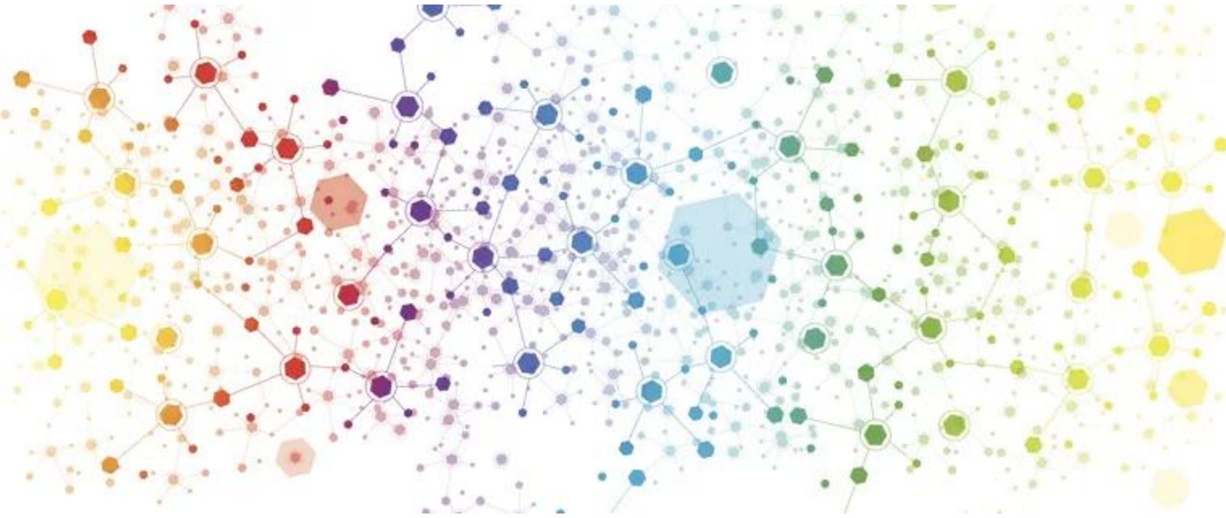
Experimental study



Deliverable 2

Group 2:

KANIAN Nicolas | MASSON Louison | CANPOLAT Guluzar | NABADJA Richard

Table des matières

EXPERIMENTAL VALIDATION

BENCHMARKS

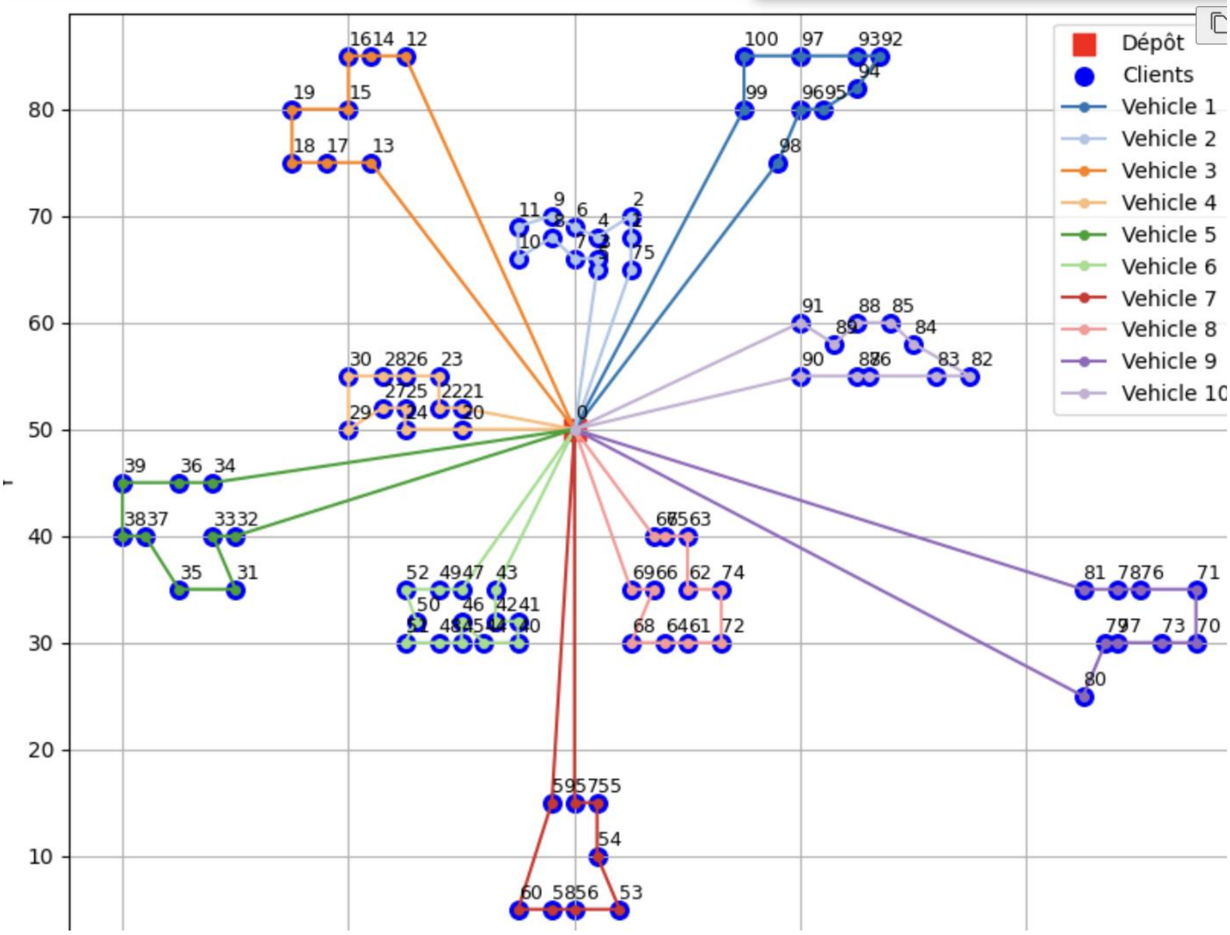
To evaluate the performance of our Tabu Search algorithm for the Vehicle Routing Problem with Time Windows (VRPTW), we conducted a series of tests using well-known Solomon benchmark instances. These instances provide a reliable and widely accepted framework for assessing the efficiency and quality of heuristic and metaheuristic algorithms in solving routing problems with time constraints.

The following table presents the results obtained from our implementation compared to the known optimal (or best-known) solutions for each instance. The Gap (%) measures the relative deviation of our solution cost from the optimal cost, while the Time (s) column indicates the computational time required to reach the reported solution.

100 clients

Instances	Optimum cost	Our cost	Gap (%)	Time
C101	828.936	828.936	0.00	39s
R101	1642.88	1701.22	3.55	47s
C104	824.78	829.69	0.60	42s
C208	588.323	591.172	0.48	45
RC201	1265.56	1369.29	8.20	1m08s
Mean	1030.49694	1064.8618	2.56	48s

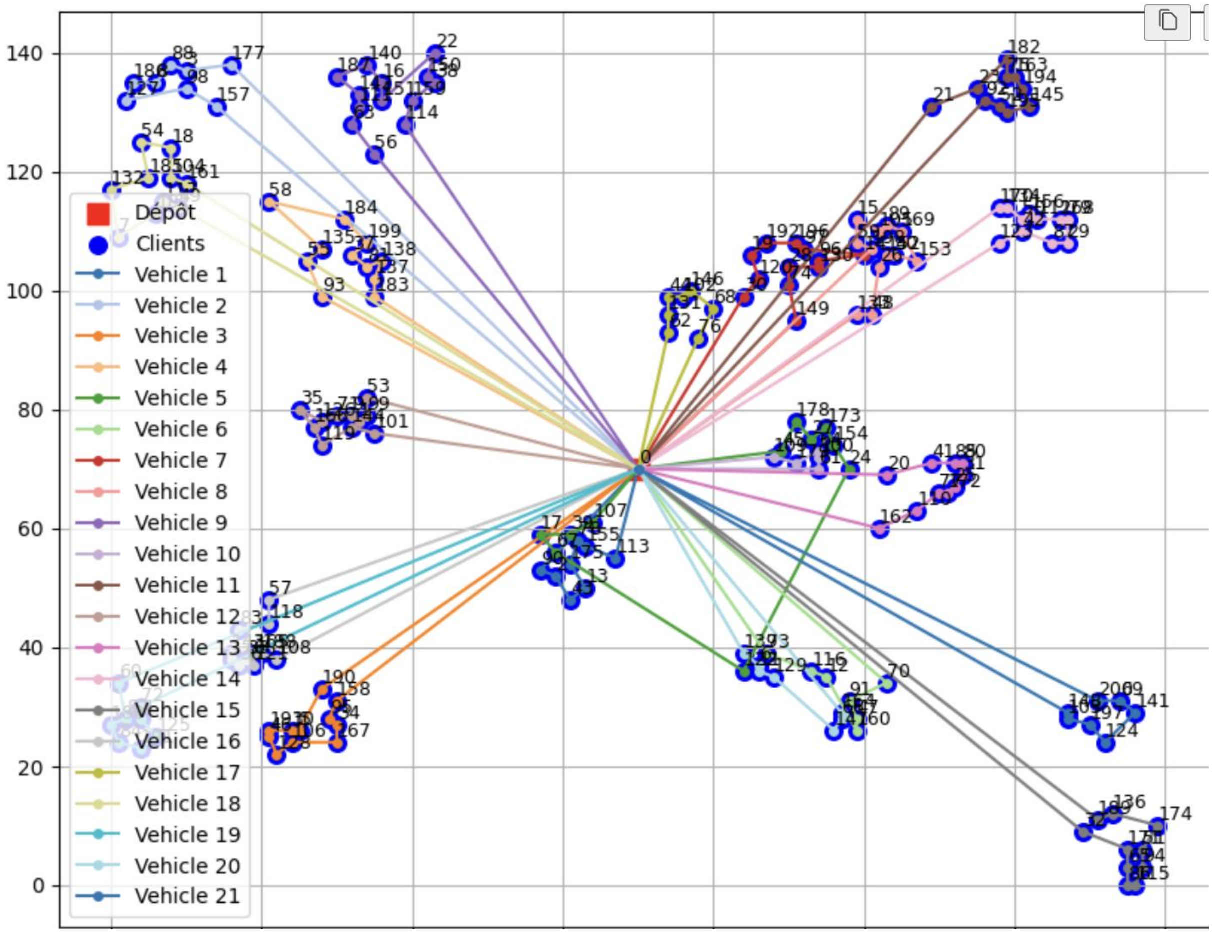
<u>Exemple graph for 100clients (C101)</u>



200 clients

Instances	Optimum cost	Our cost	Gap (%)	Time(s)
C1_2_1	2704.57	2825.73	4.48	1m55s
C1_2_2	2700.65	2749.13	1.79	1m33s
C1_2_3	2681.96	2853.60	6.40	2m53s
Mean	2695.73	2809.49	4.22	2m07s

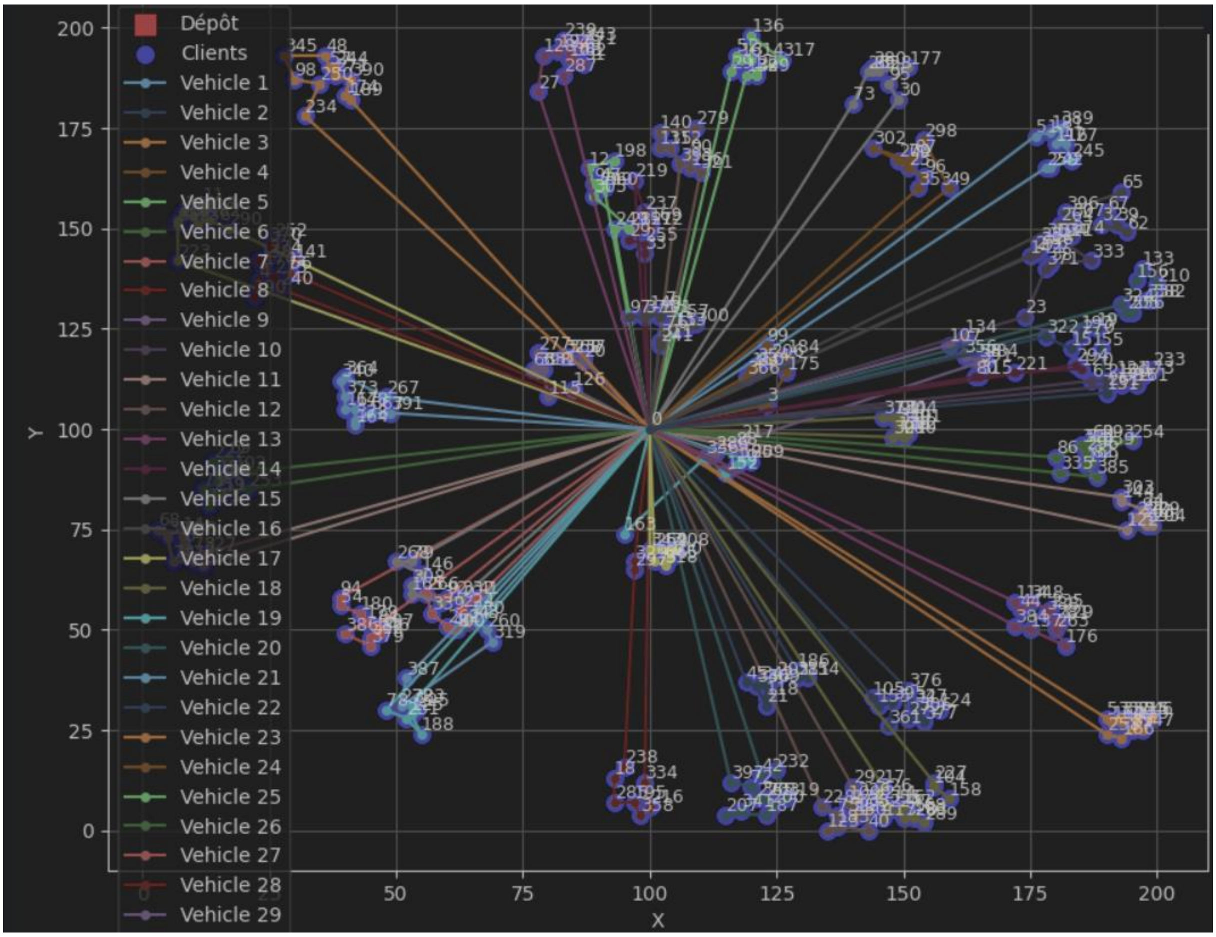
Exemple graph for 200 clients (C1_2_1)



400 clients

Instances	Optimum cost	Our cost	Gap (%)	Time(s)
C1_4_1	7152.06	7532.51	5.32	25m

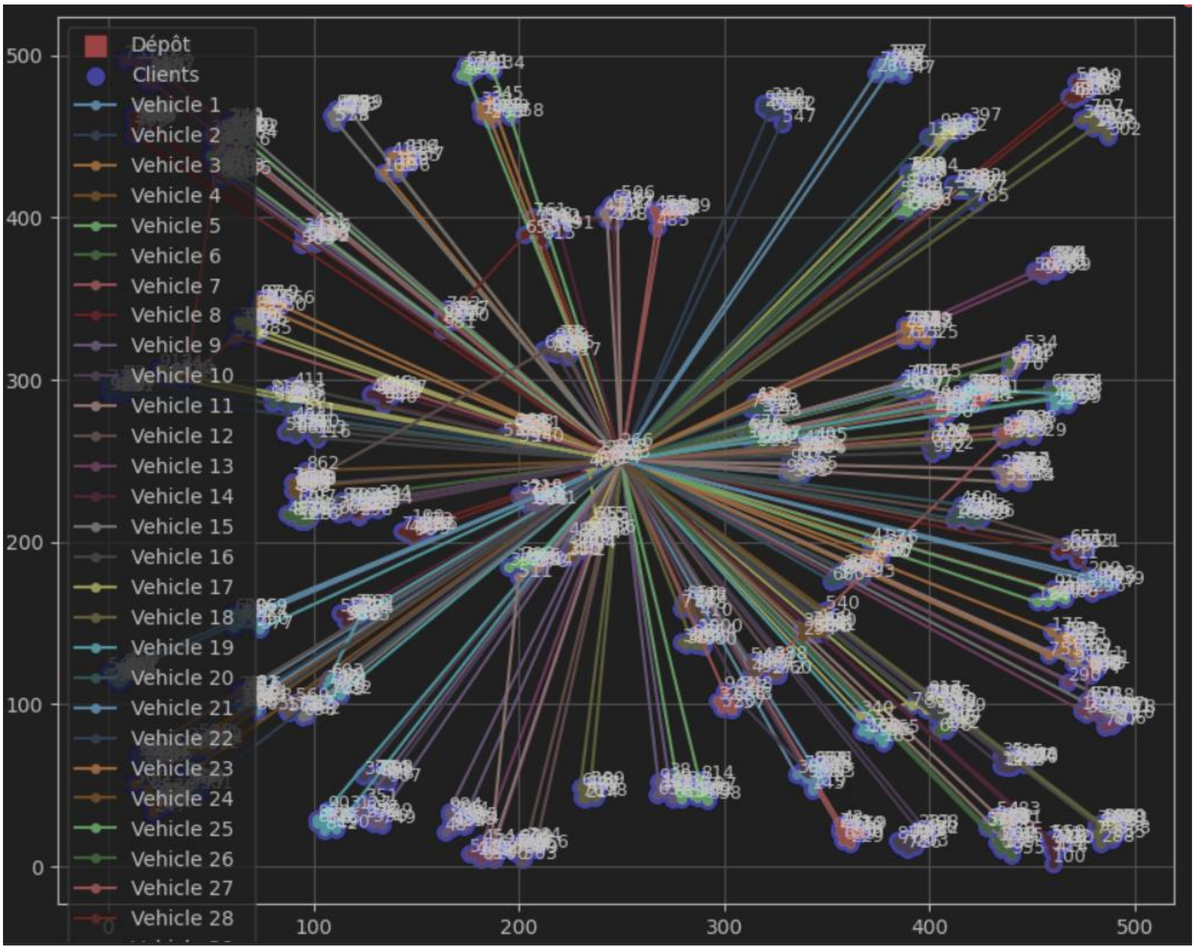
Exemple graph for 400 clients (C1_4_1)



1000 clients

Instances	Optimum cost	Our cost	Gap (%)	Time(s)
C1_10_1	42479.08	488849.74	15.00	4h

<u>Exemple graph for 1000 clients (C1_10_1)</u>



Overall, the algorithm achieved competitive results, with an average gap of 2.56% for 100 clients instances, 4.22% for 200 clients, demonstrating the effectiveness of our Tabu Search approach in finding near-optimal solutions within reasonable computational times.

However, for the RC201 instance, the gap reached 8.20%, which is relatively higher than for other cases. This deviation can be explained by the greater complexity of the RC201 instance, which combines both clustered and randomly distributed customers, making it more challenging to optimize. Moreover, this result suggests that our Tabu Search parameters, such as local_move, global_move, exploration_ratio, nb_neighbors, could be better fine-tuned to improve performance on mixed-type instances.

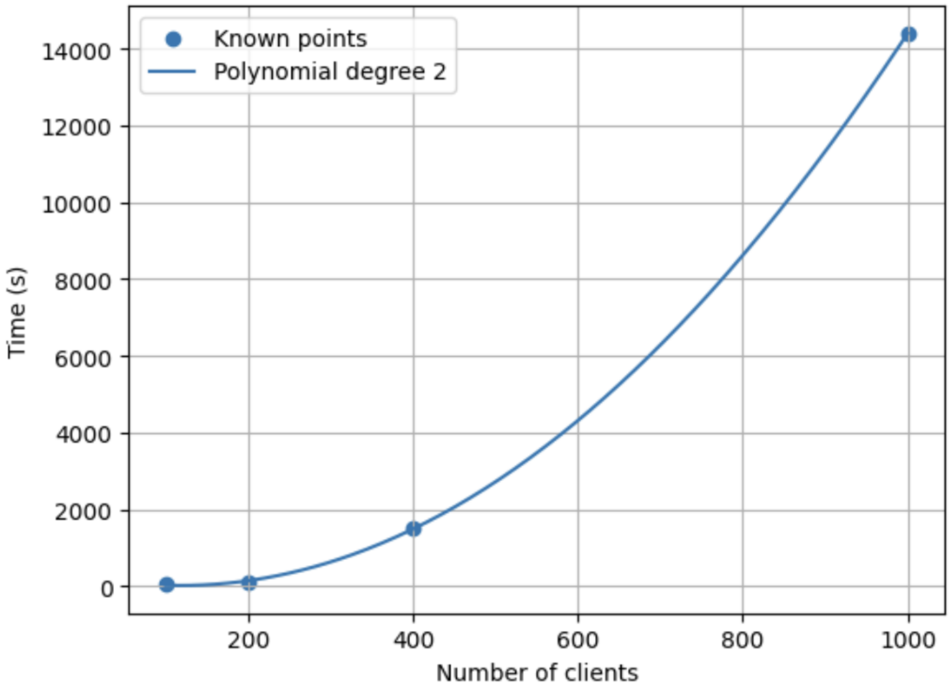
PERFORMANCE ANALYSIS

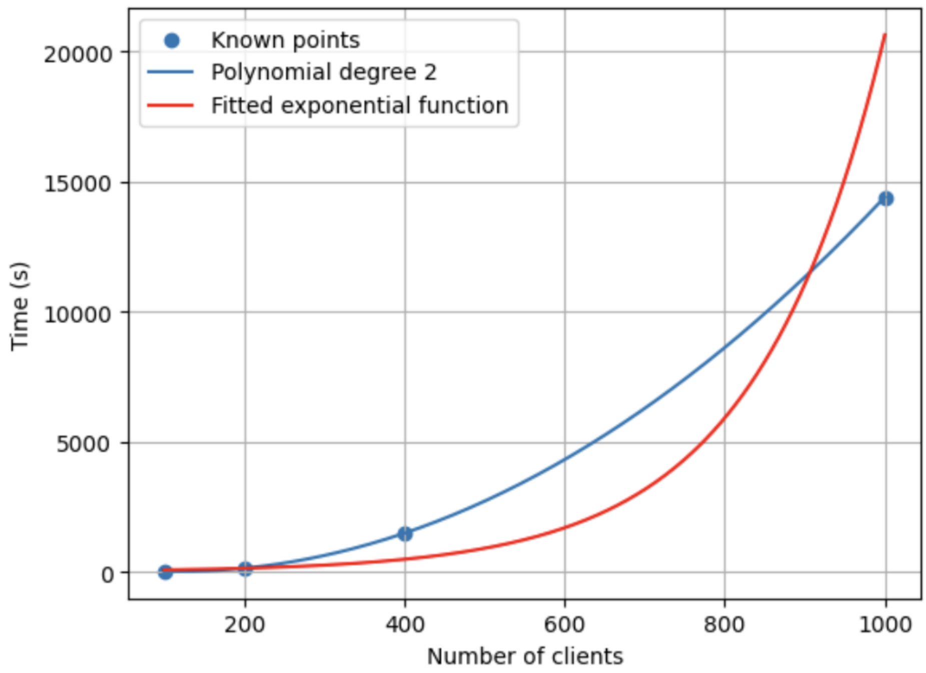
Numbers of clients	Average time	Average Gap (%)
100	48s	2.56
200	2m07s	4.22
400	25m	5.32
1000	4h	15.00

With these values we can create this function equation:

$$\boldsymbol{f(x) = 0.01849\,x^2 - 4.366\,x + 277.6}$$

And draw the curve:





COMPARISON BETWEEN THE POLYNOMIAL CURVE AND THE EXPONENTIAL FUNCTION

To analyze the evolution of computation time as the number of customers increases, we fitted two types of models to our data:

1. A **polynomial function of degree 2**, and
2. A **generic exponential function**.

Polynomial model

The polynomial regression produced the following function:

$$f(x) = 0.01849\,x^2 \;-\; 4.366\,x \;+\; 277.6$$

This is a **quadratic function**, meaning the growth rate follows an order of $O(x^2)$.

Comparison with an exponential function

When plotted alongside a classical exponential curve e^x, the differences are very clear:

- The exponential function grows **extremely fast**, becoming enormous even for moderate values of x.
- In contrast, our polynomial curve grows much more gradually.
- Even though the computation time increases as the number of customers grows, the trend remains **significantly below exponential growth**.

Interpretation

This comparison is encouraging:

- A **quadratic trend** is far less explosive than an exponential one.
- The runtime does not increase in a catastrophic way as the problem size grows.
- Staying below exponential growth means that, despite the complexity of the VRPTW problem,
  the algorithm demonstrates **reasonable scalability** for the tested instance sizes.

Conclusion

Overall, the observed polynomial behavior indicates that the computational growth remains manageable and does **not** follow the extremely rapid escalation associated with exponential functions. From a performance perspective, this is a positive and reassuring result.

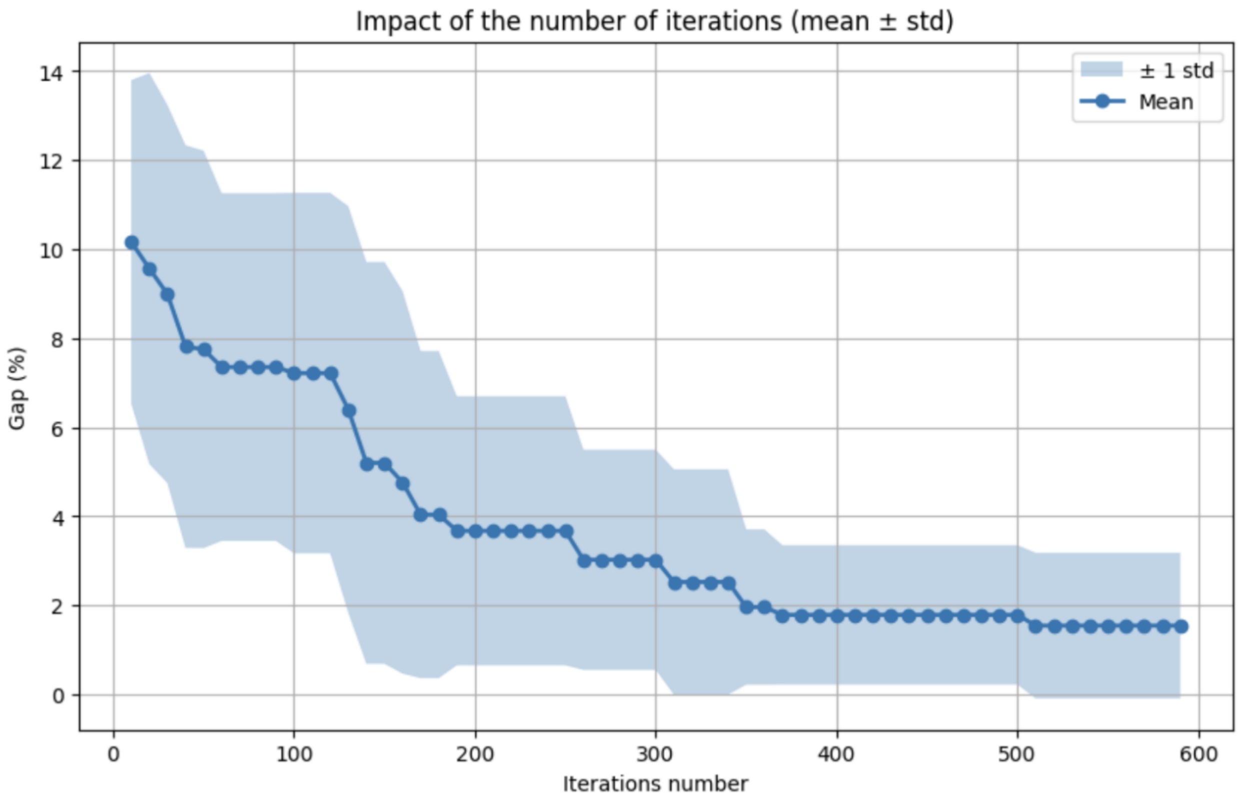
CONVERGENCE CURVES

In this section, we analyze the convergence behavior of our algorithms (Tabu Search and VNS) when solving the VRP with Time Windows. The convergence curves show how the objective value evolves over iterations, allowing us to evaluate the speed and stability of the search process. By varying key parameters (such as the tabu list size, the number of neighbors explored, or the neighborhood structures in VNS) we can observe their impact on solution quality and convergence rate. These results help identify the most effective parameter settings for achieving good performance.

All convergence analyses were performed on the same benchmark instance, **C101** from the Solomon dataset. This instance was selected because it is a well-known and widely used reference for the VRP with Time Windows, allowing for consistent comparisons between different parameter settings and algorithms. By keeping the same instance, we can clearly isolate and evaluate the effect of parameters such as the number of iterations, the tabu list size, or the number of neighbors on the convergence behavior.

NUMBER OF ITERATIONS – CONVERGENCE ANALYSIS

Impact on the gap

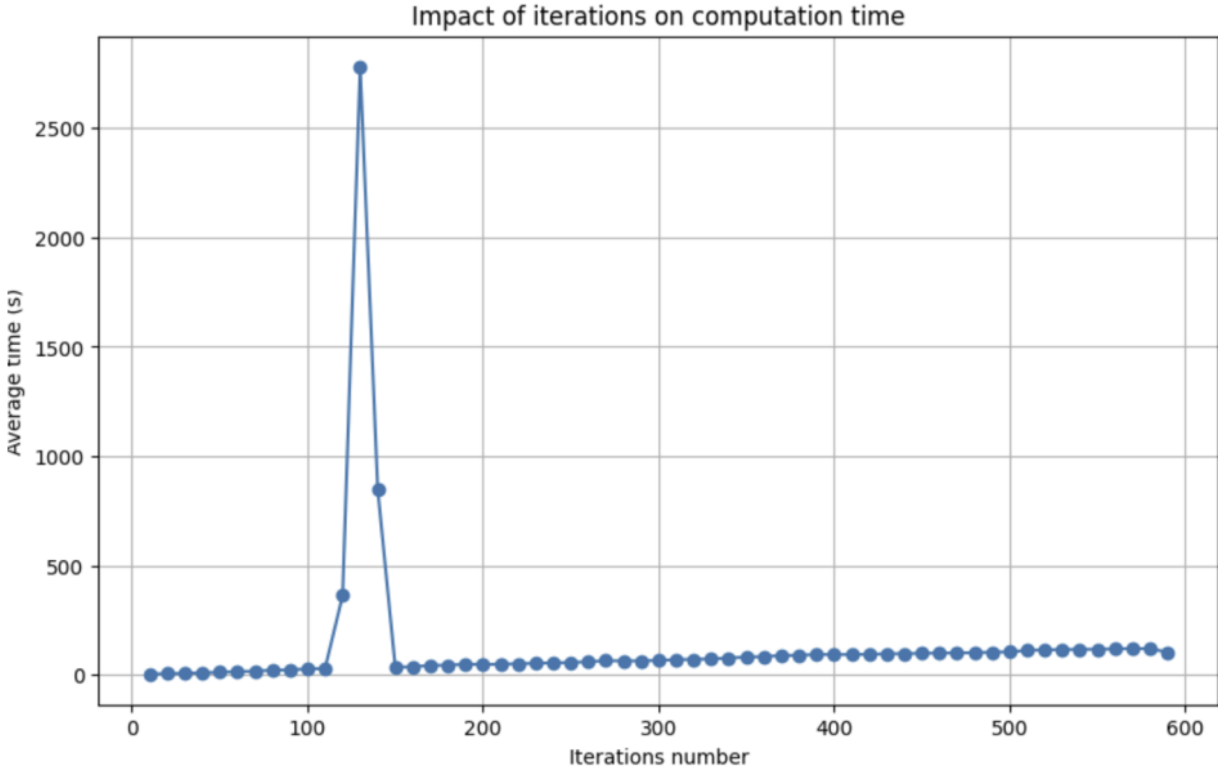


The convergence curve shows a clear decrease in the gap during the first iterations. From iteration 0 to around 350, the gap drops significantly from **10% to 2%**, indicating that most of the improvement occurs in the early phase of the search.

After 350 iterations, the curve becomes almost flat, showing **a stagnation in performance**. Between iterations 350 and 500, there is only a **very small improvement of about 0.3%**, and then the gap remains stable until the end of the experiment (600 iterations).

This suggests that the algorithm **quickly converges towards a near-optimal region**, and additional iterations bring only marginal improvements. The process could potentially achieve slightly better results with more iterations, but this would require **significantly longer computation times**, making it less efficient in practice.

Impact on time

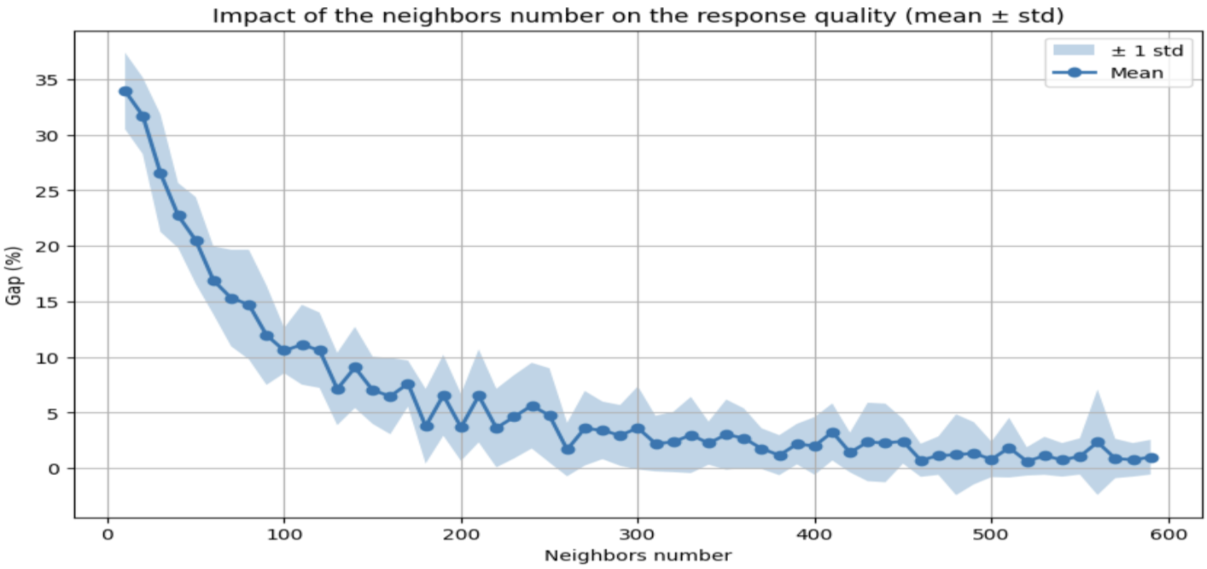


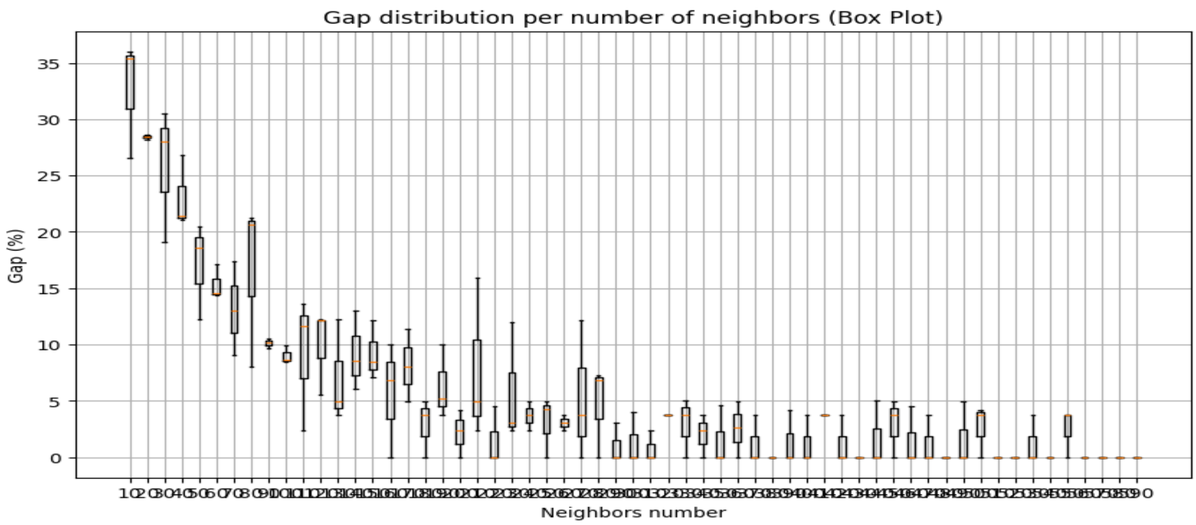
The computation time increases almost linearly with the number of iterations, reaching about **100 seconds at 600 iterations**. This linear trend indicates that each iteration has a relatively constant computational cost, meaning that extending the number of iterations directly increases the total execution time.

PS: The large peak is due to my computer going into sleep mode.

NUMBER OF NEIGHBORS – CONVERGENCE ANALYSIS

Impact on the gap

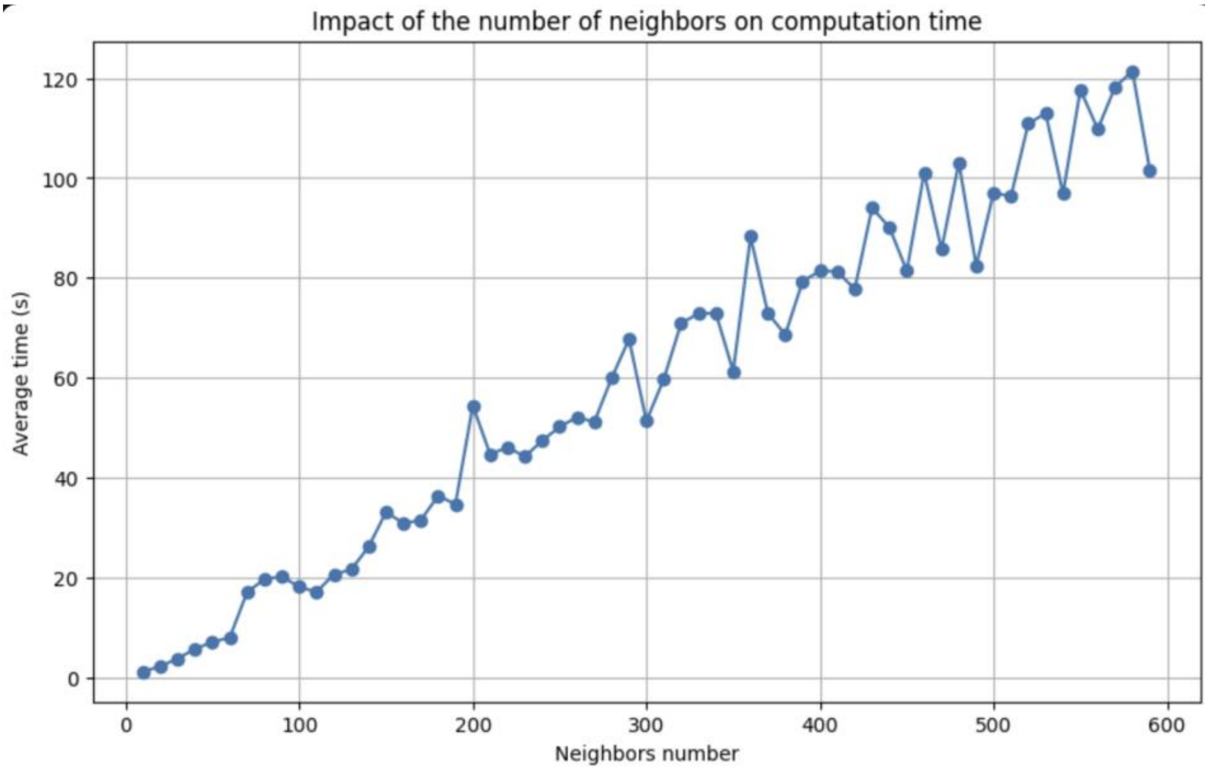




The graph shows that increasing the number of neighbors significantly improves the solution quality. The gap decreases rapidly from about **35% to below 10%** as the number of neighbors grows up to around **150**, then continues to decline more gradually until it stabilizes around **0–2%** beyond **400 neighbors**.

This indicates that exploring a larger neighborhood allows the algorithm to find better solutions, but the improvement becomes marginal after a certain point. Beyond that threshold, the gain in quality is minimal compared to the additional computational effort required.

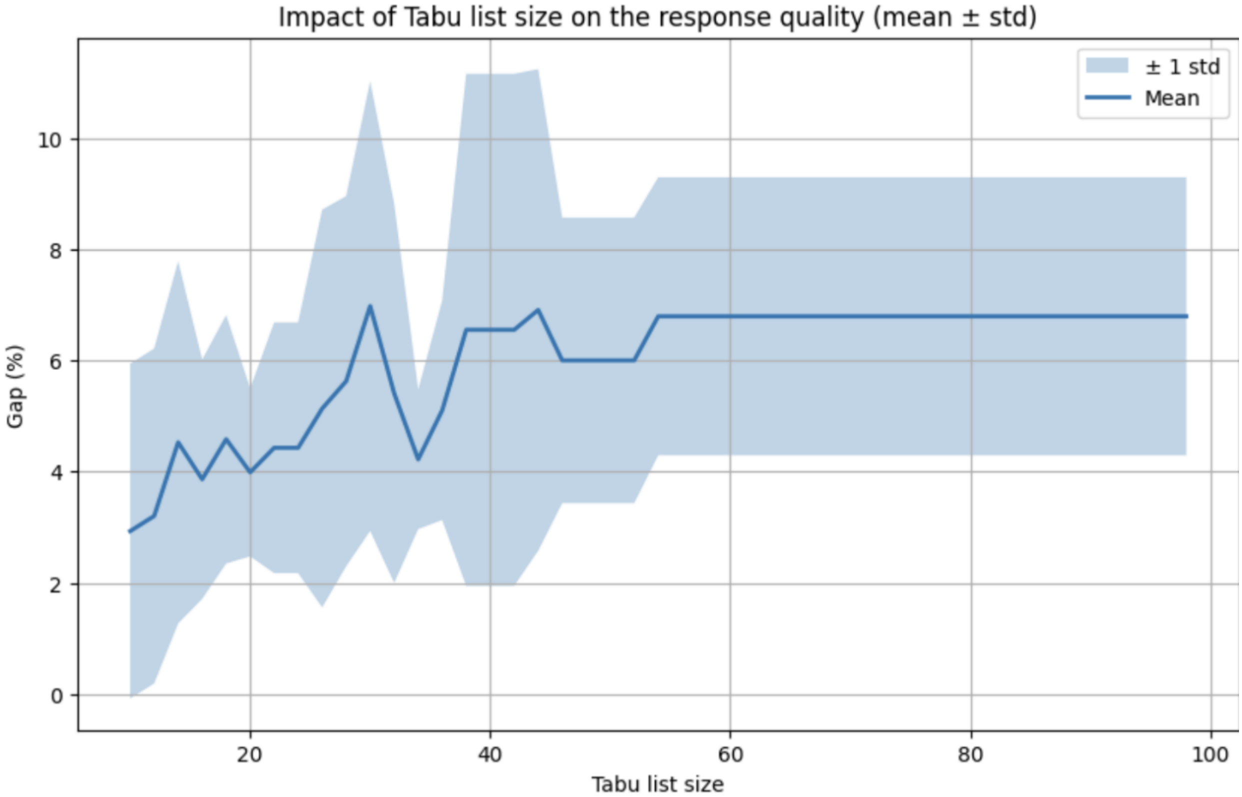
Impact on time



The convergence curve shows that the computation time increases almost linearly with the number of neighbors. For 0 to 600 clients, the runtime grows from 0 to 120 seconds, indicating that larger neighborhoods directly increase the time required for each iteration.

TABU LIST SIZE – CONVERGENCE ANALYSIS

Impact on the gap



The graph reveals three distinct phases in the relationship between tabu list size and solution quality:

<u>Phase 1: Rapid Improvement (size 10-25)</u>

The gap decreases sharply from ~3% to ~4% as the tabu list size increases from 10 to approximately 20. This initial improvement demonstrates that a minimal tabu list is insufficient to prevent cycling and allows the algorithm to explore more effectively.

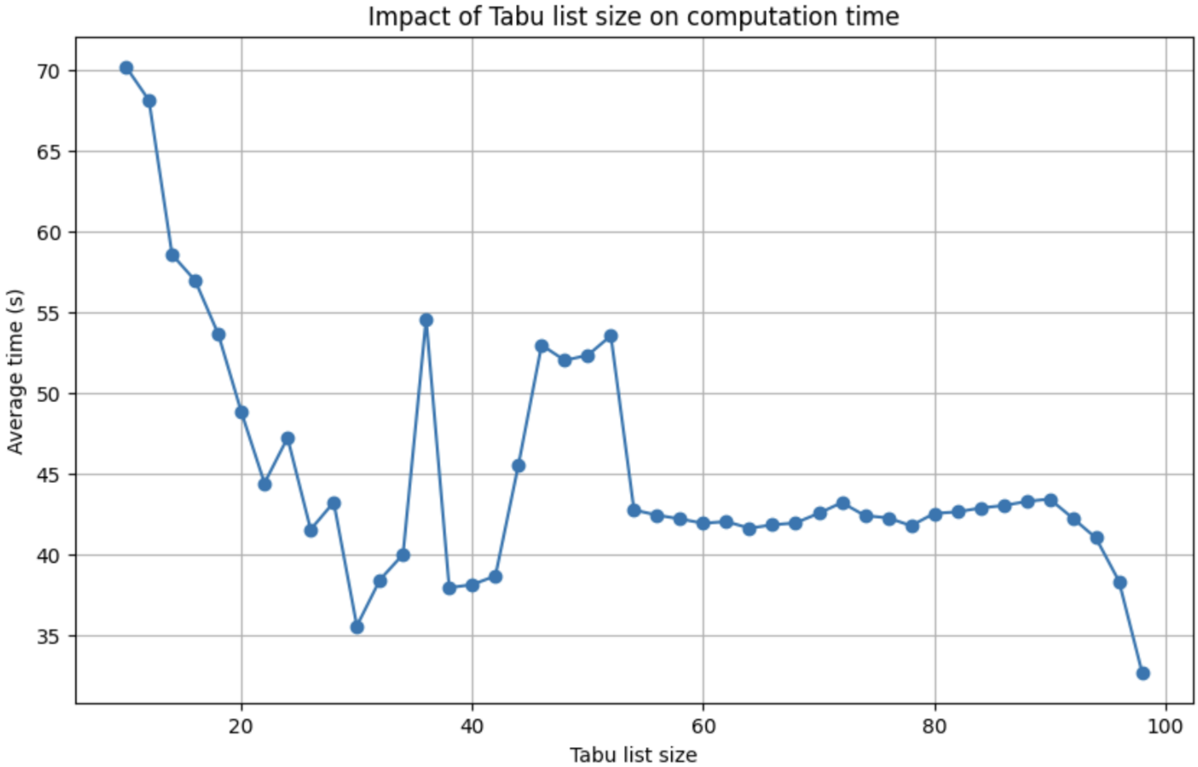
<u>Phase 2: Optimal Range (size 20-40)</u>

Between sizes 20 and 40, the algorithm achieves its best performance with gaps ranging from 4% to 7%. The relatively low variance (narrow blue band) in this region indicates consistent performance. The sweet spot appears to be around **size 15-25**, where the tabu list is large enough to prevent short-term cycling while remaining small enough to maintain search flexibility.

<u>Phase 3: Performance Degradation (size 40+)</u>

Beyond size 40, the gap stabilizes around 6-7% with significantly increased variance (wider blue band). This deterioration occurs because an overly large tabu list becomes too restrictive, prohibiting too many potentially good moves and limiting the algorithm's ability to escape local optima.

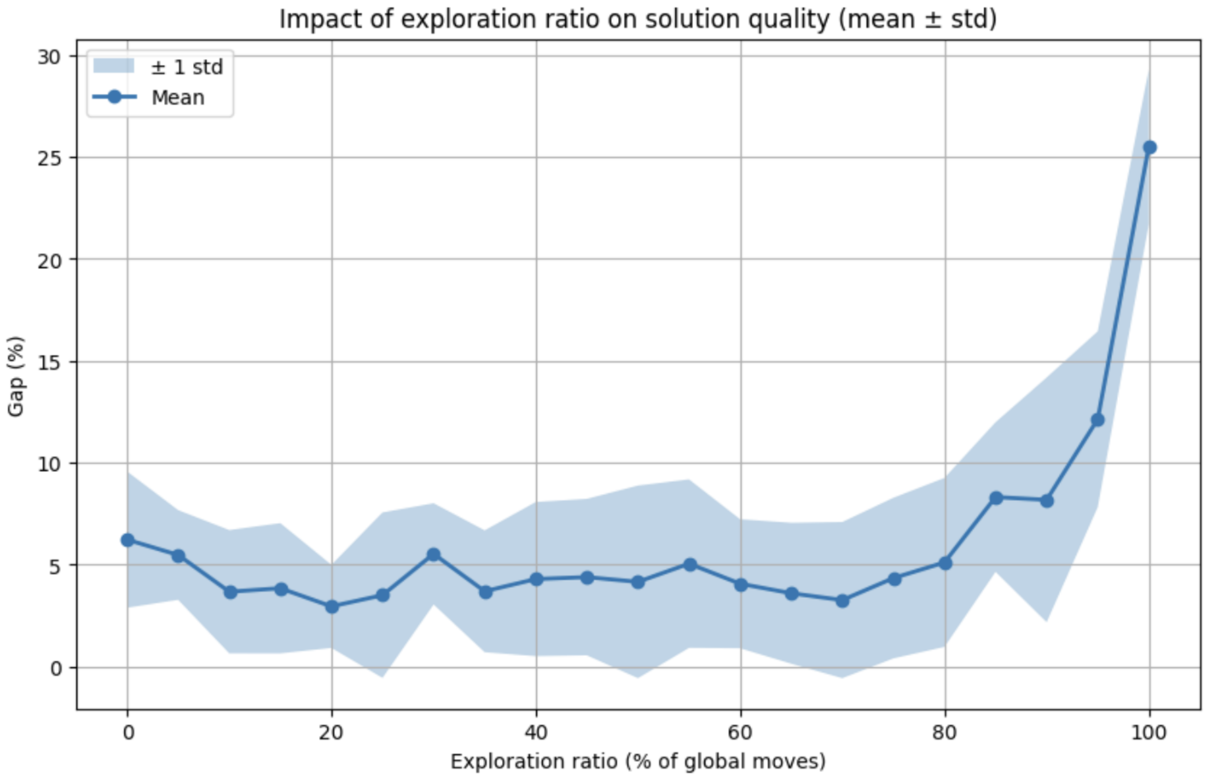
Impact on time



The computation time exhibits a strong negative correlation with tabu list size. Very small lists (size 10-15) require significantly longer execution times (~70s) due to excessive cycling that prolongs convergence. The optimal performance window occurs between sizes 15-50, achieving stable computation times around 40-55s. Beyond size 50, the time slightly decreases and stabilizes around 40s, as larger lists reduce the search space but don't significantly impact the iteration cost. The dramatic initial drop demonstrates that an adequate tabu list size is critical not only for solution quality but also for computational efficiency.

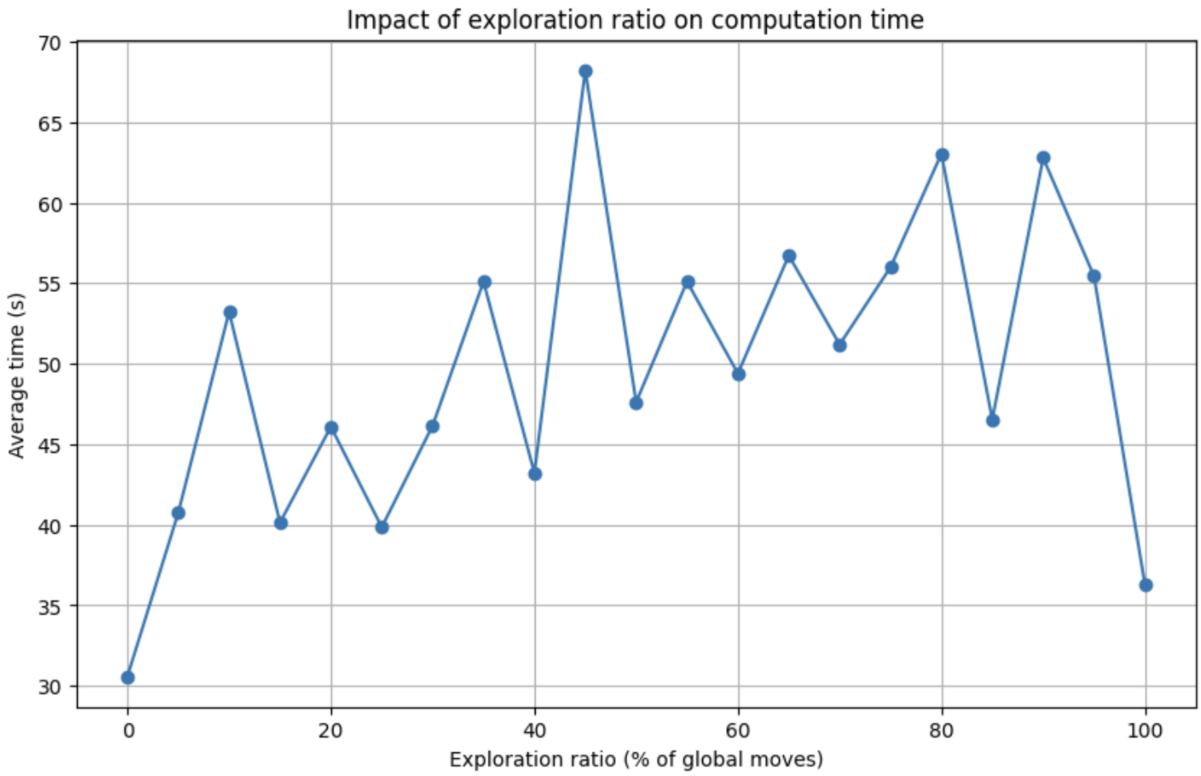
EXPLORATION RATIO – CONVERGENCE ANALYSIS

Impact on the gap



The exploration ratio demonstrates a relatively flat performance plateau between 0% and 75% global moves, with gaps consistently ranging from 3% to 6%. This stability suggests that for this specific instance (C101, 100 clients), the balance between local intensification and global diversification is remarkably flexible. However, the dramatic performance collapse beyond 80% exploration (gap jumping to 25% at 100% global moves) clearly indicates that pure exploration without local refinement is catastrophic. The optimal zone appears to be 10-20% exploration, providing the best risk-reward ratio with minimal variance. It is important to note that these results are specific to small-to-medium instances; larger instances with hundreds or thousands of clients may exhibit significantly different behavior, potentially requiring higher exploration ratios to escape complex local optima structures.

Impact on time



This curve is not interesting