A Clustering-Based Max-SAT Hybrid Approach for the Capacitated Vehicle Routing Problem with Time Windows

Pedro H. F. da Silva[1], Francisco D. C. da Silva[2], Rafael F. Dantas[2], Carlos V. D. Araújo[3], Alexandre M. Arruda[2]

[1]Instituto de Matemática, Estatística e Ciência da Computação (IME)
Universidade de São Paulo (USP) – São Paulo, SP – Brazil

[2]Universidade Federal do Ceará (UFC)
Russas, CE – Brazil

[3]Universidade Estadual do Ceará (UECE)
Fortaleza, CE – Brazil

pedrohferreira@ime.usp.br, {denilsoncordeiro, rafaeldantas}@alu.ufc.br,

carlos.vinicio@aluno.uece.br, alexandre.arruda@ufc.br

Abstract. *This paper proposes a hybrid approach for the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) that combines k-means clustering and SAT-based optimization to improve solution quality on benchmark instances. Clustering decomposes the problem into smaller routing subproblems. For each cluster, an initial feasible route is generated and subsequently improved using the 2-Opt local search heuristic. The search space is then expanded using the k-nearest neighbors algorithm, and the solution is finally refined using a SAT-based optimization model. Experiments on the Solomon benchmark instances show competitive improvements compared with a state-of-the-art approach, highlighting the potential of this strategy for CVRPTW.*

1. Introduction

The Capacitated Vehicle Routing Problem with Time Windows (CVRPTW) is a well-known combinatorial optimization problem closely related to the Traveling Salesman Problem (TSP). It is NP-hard, and as such no exact polynomial-time algorithm is known [Nanda Kumar and Panneerselvam 2012]. Most approaches in the literature focus on finding feasible high-quality solutions quickly, typically using heuristics and metaheuristics, learning-based methods, or traditional optimization formulations [Liu et al. 2023].

This paper presents an approach for the CVRPTW that combines clustering, local search and Boolean satisfiability (SAT), aiming to balance the computational efficiency of heuristic methods with the solution quality typically of exact approaches. The objective is to construct feasible routes that minimize total travel distance while satisfying capacity and time-window constraints. A modified k-means algorithm is used to partition customers, reducing the search space and improving scalability. The 2-Opt local search heuristic is then applied to refine the initial solution. Then a CVRPTW Max-SAT formulation, solved with a modern SAT-based optimizer, is used to further improve solution quality. This work extends [Ferreira and Arruda 2025], which addressed the variant without time windows using a combination of heuristics followed by Max-SAT optimization.

The source code for the project presented in this paper is publicly available on GitHub[1].

2. Problem definition and proposed solution

The TSP consists of a single salesman who starts from an initial node, visits each other node exactly once, and returns to the origin. Although the TSP has numerous real-world applications, it fails to capture several practical aspects of logistics that are critical in industrial settings. Some of these characteristics are addressed by the CVRPTW, one of its well-known variants.

The CVRPTW is defined on a complete directed graph $G = (N, E)$, where $N = \{0, 1, \ldots, n\}$ is the set of all nodes. Node 0 denotes the depot, and the remaining nodes define the customer set $C = \{1, \ldots, n\} \subset N$. A fleet of vehicles $V = \{1, \ldots, m\}$ is available to serve all customers. A solution is feasible only if the following constraints are satisfied: (i) each customer $i \in C$ has demand d_i, and each vehicle $v \in V$ has capacity Q, such that $\sum_{i \in C} d_i \cdot t_{i,v} \leq Q,\ \forall v \in V$, where $t_{i,v}$ is a binary variable equal to 1 if customer i is served by vehicle v; (ii) each customer is visited exactly once, and route continuity is enforced by the routing variable $w_{i,j,v}$ through $\sum_{v \in V} \sum_{j \in N, j \neq i} w_{i,j,v} = 1,\ \forall i \in C$; (iii) each node i has service time p_i and time window $[e_i, l_i]$, with service start time constrained by $e_i \leq S_i \leq l_i,\ \forall i \in N$. If a vehicle travels from i to j, temporal consistency is enforced using a Big-M formulation: $S_i + p_i + \text{dist}_{i,j} \leq S_j + M_{i,j}(1 - x_{i,j})$, where $x_{i,j} = \sum_{v \in V} w_{i,j,v}$. The parameter $M_{i,j} = p_i + \text{dist}_{i,j} + l_i - e_j$ is derived from the time-window bounds, ensuring that the constraint is active only when arc (i, j) is used and non-restrictive otherwise. Among all feasible solutions, the objective is to minimize the total travel distance: $min \sum_{v \in V} \sum_{(i,j) \in E} \text{dist}_{i,j} \cdot w_{i,j,v}$.

Because the CVRPTW is NP-hard, routing all customers simultaneously quickly increases the number of variables and constraints. Therefore, the modified k-means presented in Algorithm 1 is used to build feasible clusters before Max-SAT optimization. It starts by sorting customers by l_i with e_i as a tie-breaker. The initial number of clusters is set to the minimum number of vehicles needed to serve the total demand: $|V|_{\min} = \left\lceil \frac{\sum_{i \in C} d_i}{Q} \right\rceil$. This decomposition reduces the optimization to a set of smaller subproblems whose total size is $\mathcal{O}(\Sigma_{c=1}^{k} n_c^2 |V_c|)$, where n_c^2 corresponds to the number of possible customer-to-customer arcs within cluster c, which is typically smaller than $\mathcal{O}(n^2|V|)$ when clusters are non-trivial and vehicles are distributed across them. The modified k-means runs for at most 300 iterations. At each iteration, each customer is assigned to the nearest cluster that satisfies: (i) vehicle capacity, (ii) arrival before l_i, and (iii) feasible return to the depot within l_0.

Each cluster defines one routing subproblem. An initial route is generated and improved with 2-Opt [Croes 1958]. After that, the search space for the subproblem is expanded using k-nearest neighbors with $K = 5$ [Cover and Hart 1967], and the resulting candidate set is passed to the SAT stage. Service start times are encoded in binary to enable a pseudo-Boolean formulation: $S_i = \sum_{b=0}^{\lceil \log_2 l_0 \rceil + 1} 2^b S_{i,b}, \quad S_{i,b} \in \{0, 1\}$. This encoding allows time window and sequencing constraints to be expressed linearly over Boolean variables.

Some routing variables $w_{i,j,v}$ can be fixed in advance using time-window feasi-

[1] https://github.com/PedroHenriqueFerreira/CVRPTW

Algorithm 1: K-means Clustering for CVRPTW

Sort customers by (l_i, e_i);
Initialize clusters and centroids;
for $t = 1$ **to** 300 **do**
  Clear all clusters;
  for *each customer* i **do**
    Assign i to best feasible cluster; otherwise, add to unassigned set;
  for *each unassigned customer* i **do**
    Insert i into nearest feasible cluster; otherwise, create new cluster;
  Update centroids to mean coordinates of assigned customers;
  if *centroids unchanged* **then**
    break;
return clusters;

bility. The earliest arrival at customer j is obtained by departing from customer i at its ready time and adding $p_i + \text{dist}_{i,j}$, where $\text{dist}_{i,j}$ is the Euclidean distance between i and j. If this arrival exceeds l_j, arc (i, j) is infeasible and $w_{i,j,v}$ is fixed to zero: $e_i + p_i + \text{dist}_{i,j} > l_j \implies w_{i,j,v} = 0$. Finally, the return to the depot must also respect the depot time window. Therefore, after servicing the last customer k of that route, the arrival time at the depot must satisfy: $S_k + p_k + \text{dist}_{k,0} \leq l_0$.

3. Results and discussion

Using the Solomon benchmark instances [SINTEF 2022], the proposed methodology was evaluated, and the results are summarized in Table 1. The Best Distance column reports the best-known distance for each instance as reported in the benchmark. The RL + RO + SAT column presents the best distance and its gap to the benchmark found by the state-of-the-art approach of [Khadilkar 2022], which combines reinforcement learning, rollouts, and SAT. The third column reports the results obtained using only k-means and 2-Opt, while the last column shows the results when SAT is also integrated. For each instance size, the last row reports the mean runtime of each approach.

Both distance optimization and execution time were treated as equally important considerations in the algorithm design. Experiments were run on a computer equipped with an Intel Core i5-13420H (4.60 GHz, 8 cores) and 8 GB of RAM, using the Ubuntu 25.10 operating system. The implementation was written in Python 3.13 and used the *clasp* solver [Gebser et al. 2007]. With those specifications, the mean runtimes were considerably lower than the computational times reported in the state-of-the-art results.

As suggested by [Le et al. 2022], k-means alone already provides good solutions within a few seconds, while the addition of SAT improves solution quality at the cost of increased computational time. In particular, incorporating SAT reduces the mean gap to the best-known solution from 29.4% to 26.0%, but increases the mean runtime from 0.2 s to 21.1 s across the aggregated instances presented in Table 1. In line with this trade-off, the proposed k-means+2Opt+SAT outperforms the k-means+2Opt baseline at all evaluated scales, except in terms of runtime, achieving average distance reductions between

1.91% and 3.84%. These improvements are statistically significant according to both the paired t-test and the Wilcoxon signed-rank test ($p < 0.05$), providing consistent evidence against the null hypothesis under both parametric and non-parametric assumptions.

Although our method achieved shorter distances in several individual instances, the state-of-the-art approach in [Khadilkar 2022] still reports a lower overall distance and generally fewer vehicles across the entire dataset, with a 24.2% mean gap to the best-known solutions. Even so, our results offer meaningful partial improvements that can inform future work. When summing the distance differences for the 100-customer instances between the proposed methodology and the state-of-the-art approach, a 2.6% reduction in traveled distance was observed. This may indicate a behavioral trend in which the algorithm performs better on denser instances. In addition, for the 25 and 50 customer instances, our approach consistently required fewer vehicles.

Table 1. Comparison of results with state-of-the-art solutions

Data	Best Dist	RL+RO+SAT		k-means + 2Opt		k-means + 2Opt + SAT	
		Dist	Gap	Dist	Gap	Dist	Gap
C1-25	191	**192**	**0.5%**	200	4.7%	200	4.7%
R1-25	464	568	22.4%	547	17.8%	**526**	**13.3%**
RC1-25	350	**362**	**3.4%**	377	7.7%	376	7.4%
C2-25	216	261	20.8%	273	26.3%	**260**	**20.3%**
R2-25	382	513	34.2%	516	35.0%	**447**	**17.0%**
RC2-25	319	**329**	**3.1%**	442	38.5%	436	36.6%
Time (s)	*(-)*	*(15.00s)*		***(0.01s)***		*(3.54s)*	
C1-50	362	**374**	**3.3%**	407	12.4%	400	10.4%
R1-50	766	963	25.7%	924	20.6%	**907**	**18.4%**
RC1-50	730	**849**	**16.3%**	908	24.3%	893	22.3%
C2-50	357	525	47.0%	509	42.5%	**499**	**39.7%**
R2-50	634	824	29.9%	855	34.8%	**823**	**29.8%**
RC2-50	585	**631**	**7.8%**	932	59.3%	923	57.7%
Time (s)	*(-)*	*(60.00s)*		***(0.11s)***		*(18.50s)*	
C1-100	826	**869**	**5.2%**	1013	22.6%	1009	22.1%
R1-100	1210	1646	36.0%	1545	27.6%	**1525**	**26.0%**
RC1-100	1384	1788	29.1%	1779	28.5%	**1755**	**26.8%**
C2-100	587	1012	72.4%	818	39.3%	**795**	**35.4%**
R2-100	902	1309	45.1%	1258	39.4%	**1216**	**34.8%**
RC2-100	1063	**1427**	**35.2%**	1576	48.2%	1544	45.2%
Time (s)	*(-)*	*(350.00s)*		***(0.49s)***		*(41.30s)*	

4. Concluding remarks

This paper presented a compromise between a heuristic-only approach, which is extremely fast but yields suboptimal distances, and reinforcement-learning-based methods, which achieve higher solution quality at a greater computational cost. The results show that integrating SAT with k-means and 2-Opt consistently improves solution quality, reducing the mean optimality gap from 29.4% to 26.0%, while maintaining relatively low runtimes compared to state-of-the-art approaches.

References

Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812.

Ferreira, P. and Arruda, A. (2025). Post-improving the capacitated vehicle routing problem using a max-sat solver. In *Anais do VI Workshop Brasileiro de Lógica*, pages 32–39, Porto Alegre, RS, Brasil. SBC.

Gebser, M., Kaufmann, B., Neumann, A., and Schaub, T. (2007). clasp: A conflict-driven answer set solver. In Baral, C., Brewka, G., and Schlipf, J., editors, *Logic Programming and Nonmonotonic Reasoning*, pages 260–265, Berlin, Heidelberg. Springer Berlin Heidelberg.

Khadilkar, H. (2022). Solving the capacitated vehicle routing problem with timing windows using rollouts and max-sat.

Le, T. D. C., Nguyen, D. D., Oláh, J., and Pakurár, M. (2022). Clustering algorithm for a vehicle routing problem with time windows. *Transport*, 37(1):17–27.

Liu, F., Lu, C., Gui, L., Zhang, Q., Tong, X., and Yuan, M. (2023). Heuristics for vehicle routing problem: A survey and recent advances.

Nanda Kumar, S. and Panneerselvam, R. (2012). A survey on the vehicle routing problem and its variants. *Intelligent Information Management*, 04.

SINTEF (2022). Solomon benchmark. `https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/`. Accessed: Mar 2026.