 doi:10.20944/preprints202206.0074.v1

Article

A Cooperative Parallel Tabu Search Meta-heuristic for the Vehicle Routing Problem with Time Windows

Hassan El Fahim [1,]

[1] Department of Mathematics and Computer Science, Hassan II University of Casablanca, Morocco; contact.dr.elfahimhassan@gmail.com

Abstract: This paper presents a cooperative parallel tabu search meta-heuristic for the vehicle routing problem with time windows. It is based on the scheme in which several search threads cooperate by asynchronously exchanging information on the best solutions identified. The exchanges are performed through a mechanism called adaptive memory which holds and manages a pool of solutions. This enforces the asynchronous strategy of information exchanges and ensures the independence of the individual search threads. Each of these independent threads implements a tabu search meta-heuristic. Comprehensive computational experiments and comparisons to best known solutions show that the proposed cooperative parallel tabu search algorithm is able to achieve 48 new best solutions on VRPTW benchmark instances.

Keywords: vehicle routing problem; time window; parallel meta-heuristic; cooperative search; tabu search

1. Introduction

Transportation is generally considered as being the most important economic activity among the components of a business logistics system. In fact, transportation costs account for more than 50% of the total logistics costs for many companies. This can be explained by the fact that transporting is required in the whole production process from manufacturing to delivery to the final customers and returns. Thus, the management of transportation services is clearly an important task for logistics and supply chain decision makers.
Solving the Vehicle Routing Problem (VRP) [1] is a key to efficient transportation management and logistics coordination. It may be used for strategic decisions such as the long-term acquisition of vehicles, for tactical design of fixed routes or minute to minute revision of routing plans. It has a large number of real world applications and comes in many guises depending on the objective, the type of operation, the time frame for decision making and the constraints that must be adhered to. In broad terms, it seeks an optimal routing plan for a fleet of vehicles that has to meet transportation services subject to certain operational constraints. The objective is to minimize total transportation costs. Typical operational constraints pertaining to the maximum load permitted, total duration of travel of a vehicle on any route and restrictions on when a customer may be served. In vehicle routing problem with time windows (VRPTW), service to customers must occur within an associated time intervals called time windows. That is, arriving at a customer later than latest time of its time window is strictly forbidden. Similarly, a waiting is incurred if a vehicle reaches to a customer before its earliest time window. The objective is to minimize total transportation costs while serving every customer exactly once and satisfying vehicle capacity and time window constraints on every route. In general, transportation costs are often expressed as a combination of fleet acquisition/depreciation costs and transportation costs for the routing plan.
In terms of solution methods, there is a huge amount of literature on methods proposed to deal with VRP and its variants. Research on exact methods, heuristics and meta-heuristics for solving vehicle routing problems is nowadays highly on the rise. The main advantage of using exact methods for solving vehicle routing problems is the guarantee of finding an optimal routing plan. Whereas the critical disadvantage comes from the exponential growth of the computation time according to the instance size when solving real world problems.

Citation: El Fahim, H. . *Preprints* **2022**, *1*, 0. https://doi.org/

 doi:10.20944/preprints202206.0074.v1

On the other hand, specific heuristics tend to be very fast but the solutions obtained are generally not of high quality. Furthermore, the complexity of vehicle routing problems which are often NP-hard and/or the limited resources available to solve them have made the development of meta-heuristics a major field of study. These solution methods form a powerful class of optimization techniques that have gained a lot of popularity in recent years; they can offer a tradeoff between computation time and solution quality. On the other hand, due to the fast development of computer science in recent years increasingly harder and more complex problems are being faced continuously. Although the use of meta-heuristics allows to significantly reduce the time of the search process, the high dimension of many tasks will always pose problems and result in time-consuming scenarios. Thus, even meta-heuristics may require computing times exceeding the limits of what is considered as acceptable.

Parallel computing is a type of computation in which many calculations are carried out simultaneously operating on the principle that large problems can often be divided into smaller ones which are then solved at the same time. In case of meta-heuristics, parallel computing can help not only to speed up finding good quality solutions but also to enhance the robustness of the approaches. Parallel meta-heuristics were then born as an intersection of meta-heuristics and parallel computing, but in recent decades they have been established as an independent field of study. There exist a large number of surveys and taxonomies for parallel meta-heuristics. The reader is referred to the work by [2]; we believe that is the most illustrative taxonomy for parallel meta-heuristics. Besides, we refer to the works by ([3], [4]) which can be used as a pointer to further study parallel meta-heuristics. Like other meta-heuristics, tabu search [5] has been the object of several efforts aimed at taking advantage of the benefits offered by parallel computing to deal with many optimization problems especially vehicle routing problems.

In this paper, we present a cooperative parallel tabu search meta-heuristic to deal with the VRPTW. It is based on the scheme introduced by [6] in which several search threads cooperate by asynchronously exchanging information on the best solutions identified. The exchanges are performed through a mechanism called adaptive memory which holds and manages a pool of solutions. This enforces the asynchronous strategy of information exchanges and ensures the independence of the individual search threads. Each of these independent threads implements a tabu search meta-heuristic. The rest of this paper is organized as follows: First, we present the basic concepts of tabu search meta-heuristic and the strategies used over time to design parallel tabu searches. Next, we describe the proposed cooperative parallel tabu search meta-heuristic as well as: the objective defined to evaluate a solution, the procedure implemented to generate initial solutions, the neighborhood structure used to get tabu searches moving and how tabu search processes get information from the adaptive memory. Then, we compare the proposed cooperative parallel tabu search algorithm solution quality against published results. And finally, we conclude the paper with conclusions.

2. The proposed solution method

In the following, we present the proposed solution method. The basic concepts of tabu search meta-heuristic are first described. Then, the cooperative parallel tabu search meta-heuristic is presented as well as its components.

2.1. Tabu search meta-heuristic

Tabu search is a widely used meta-heuristic for solving vehicle routing problems. Although it was introduced in 1986 by [5], it borrowed many artificial intelligence ideas suggested during the sixties. It is primarily centred on allowing to design a non-monotonic trajectory in order to guide the search to promising regions of the solution space. Indeed, tabu search moves at each iteration t from a solution S_t to its best neighbor S_{t+1}. If $f(S)$ denotes the cost of a solution S, then $f(S_{t+1})$ is not necessarily better than $f(S_t)$. However the danger with this exploration strategy is that after moving away, it is likely that the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

search will cycle right back to the local optimum just visited. To avoid cycling, tabu search forbids for a number of iterations moves that would return to a solution recently examined. Tabu search records these forbidden moves, which are referred to as tabu moves, using a short-memory called *tabu list* in tabu search terminology. On the other hand and as one can intuitively expect, it is absurd to prohibit a move that leads to a solution better than all those examined during preceding iterations. In order to avoid missing such solution which may be optimal, it is required to disregard the possible tabu status of such move. According to tabu search terminology, such exception is refereed to as *aspired move*. This use of memory to guide the search is a distinctive feature that has its roots in the field of artificial intelligence as mentioned in the beginning of this section. Another feature of tabu search is that it is particularly well suited for parallel implementation using different parallelization strategies. The strategies that have been used over time to design parallel tabu search algorithms are based on three main procedures: functional decomposition, domain decomposition and multi-search threads.

- Functional decomposition: This parallelization strategy aims to reduce the computational time of the sequential tabu search meta-heuristic. This is accomplished by executing some or all operations within an iteration of the sequential tabu search in parallel. For example, the search for the best neighbor which is known to be the most computationally expensive part of a tabu search can be executed at each iteration in parallel. Note that a master-slave paradigm is often used to implement tabu search algorithms following this parallelization scheme. In the case of evaluating a neighborhood in parallel, the master thread executes a sequential tabu search and divides at each iteration the neighborhood into subsets. Each slave thread evaluates a subset of the partitioned neighborhood assigned by the master thread. And then, each slave thread sends its best neighbor back to the master thread which determines the best neighbor among all it has received.
- Domain decomposition: In this strategy, the problem domain or search space is decomposed. Similarly, a master-slave framework is used to implement these parallel tabu search algorithms where the master thread creates partitions and assigns them to the slave threads. Typical implementation consists of partitioning the set of decision variables into subsets each of which is assigned to a slave thread. Each slave performs a tabu search and sends back the partial solution it has obtained to the master thread. The latter combines the partial solutions to create a complete solution for the main problem.
- Multi-search parallelism: This parallelization strategy consists of guiding multiple tabu searches each of which deals with the main problem. The implementation of these parallel algorithms uses a framework in which a number of independent threads work on the same solution space. It should be noted that these tabu search threads may start from the same or different initial solutions, may or may not: use a distinctive set of parameter values, communicate during the search. If these threads are totally independent and the best overall solution is returned at the end as output, the implemented approach is nothing but a parallel restarting. Otherwise, the overall search process may involve cooperation between tabu search threads that consists of exchanging information during the search. This exchange of information is implemented by creating a pool of elite solutions found by different tabu search threads. The elite solutions can be used by each tabu search thread to implement diversification and intensification strategies in the same way as in the sequential tabu search.

In the next section, we describe the proposed cooperative parallel tabu search meta-heuristic.

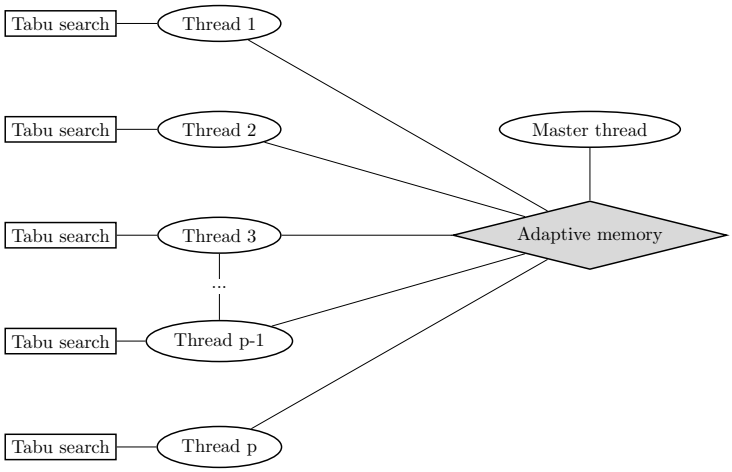
2.2. The proposed cooperative parallel tabu search meta-heuristic

In our implementation of a cooperative multi-search parallelism, each thread executes a tabu search meta-heuristic. The cooperation is achieved through asynchronous exchanges of information among tabu search threads through a common adaptive memory. In fact,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

threads exchange information about their best solutions identified so far. That is, when a thread terminates a search process it sends out the value of the obtained solution as well as the routes composing it to the adaptive memory. Similarly, when a thread desires to receive information it reaches out and takes it from the adaptive memory. Indeed, each thread uses this information differently to generate a starting solution for the next search process.
Our implementation uses a master-slave framework where the master thread manages the adaptive memory and slaves run tabu searches. First, a set of initial solutions is created each of which is improved by a tabu search. The obtained solutions are then used to initialize the adaptive memory. In the concurrent process, slave threads feed memory with new improved solutions and request new starting solutions from it. Note that a new solution replaces the worst solution in memory if it is better. The master thread manages the adaptive memory and creates initial solutions for the slave threads that run tabu search algorithms. Figure 1 illustrates the proposed cooperative parallel tabu search algorithm. In the next subsections, we present the components of the proposed cooperative parallel tabu search meta-heuristic.

Figure 1. The proposed cooperative parallel tabu search meta-heuristic



2.2.1. VRPTW objective function

A company's objective in freight transportation is usually related to the overall business goal of minimizing costs. To support this objective, the decision maker tries to minimize the total fulfillment costs for all customers service. These fulfillment costs consist of fixed and variable costs. Fixed costs arise for example from the deployment of vehicles whereas variable costs may arise from the costs for fuel and from depreciations for wear and tear. The primary objective is chosen to be the minimization of the number of vehicles needed to serve all customers. This objective supports the company's objective since by minimizing the number of vehicles, the tied-up capital for the fleet is minimized and as a consequence fixed costs are minimized. As a secondary objective, the decision maker tries to minimize the total travel distance which can be seen as a main driver of transportation costs. This objective clearly fosters the reduction of variable costs for serving customers by reducing vehicles depreciation and fuel consumption. These two objectives are widely used in the literature on vehicle routing problems with time windows. Hence the primary objective is the minimization of the number of vehicles needed to serve all customers. A secondary objective is the minimization of the total travel distance. In addition, it is assumed that freight transportation cost is symmetric, i.e., the cost for shipping freight from location i to location j is equal to cost for shipping freight from location j to location i. Also, we suppose that:

- all freight to be delivered to customers is located at a central depot.
- all vehicles are based at the depot.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

- split delivery is not allowed.
- each vehicle route starts from and ends at the depot.

2.2.2. Initial solutions

We use the insertion heuristic of [7] to generate initial solutions for tabu searches. This heuristic was found to be successful to generate good initial solutions in the present of time window constraints. In consists of constructing a route at each iteration by inserting unserved customers. At the beginning of every route construction, an empty route connecting the depot to itself is constructed and marked as the current partial route. The heuristic examines at each iteration every unserved customer for insertion by testing every feasible place on the current partial route by means of an insertion cost $c_1(i,u,j)$. This user-defined cost is expressed as follows:

$$c_1(i,u,j) = \alpha_1 \cdot (d_{iu} + d_{uj} - \mu \cdot d_{ij}) + \alpha_2 \cdot (b_{ju} - b_j) \quad (1)$$

Where u denotes an unserved customer.
d_{ij} is the distance between vertices i and j.
b_{ju} denotes the new start of service time at customer j after inserting u.
b_j denotes the original start of service time at j.
Next, the best unserved customer in terms of a second selection cost $c_2(i,u,j)$ is inserted into the current partial route. This user-defined cost is expressed as follows:

$$c_2(i,u,j) = -\lambda \cdot d_{0u} + c_1(i,u,j) \quad (2)$$

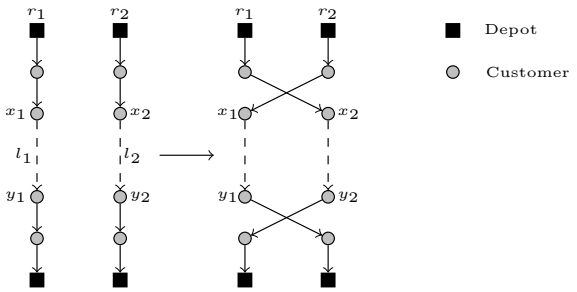
Where d_{0u} denotes the distance between the depot and customer u.
This procedure is repeated until no further customer can be inserted into the partial route without violating vehicle capacity and/or time window constraints. If there are still unserved customers, the heuristic starts the construction of a new route. This procedure of route construction is repeated until the insertion of all customers. It should be noted that the weights α_1, α_2, μ and λ determine the relative contribution of each metric in the determination of the best unserved customer at each iteration.

2.2.3. Neighborhood structure

One can describe a neighborhood structure as the main driver of a meta-heuristic. In vehicle routing problems, it is defined by all or some solutions that can be reached from the current solution by removing a set of customers from one route to another in order to make it empty and therefore to reduce the number of vehicles used to serve customers. The neighborhood structure of our tabu searches is based on the cross exchange procedure introduced by ([8], [6]). This operator generalizes both 2-opt* and Or-opt exchanges. The neighborhood of a solution S according to cross exchange is obtained by exchanging subsegments of customers between routes. This neighborhood structure restricts the subsets of customers chosen in each route to be consecutive. For each pair of routes r_1 and r_2, we randomly generate two lengths l_1 and l_2. Then, we test the exchange of every segment of length l_1 on route r_1 and every segment of length l_2 on route r_2 while respecting the orientation of the routes. The best feasible exchange in terms of time windows and vehicle capacity is recorded. This procedure is applied for each pair of routes of S. The best overall exchange is recorded and the resulting solution is marked as the best neighbor of S. In addition, the two sequences of customers involved in the best overall exchange are marked, for a number of iterations, as tabu. An illustration of this neighborhood structure is given by Figure 2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Figure 2. Cross-exchange neighborhood structure



2.2.4. Adaptive memory

The adaptive memory is used as a repository of elite solutions. It stores previously found solutions and uses them to generate new starting solutions for tabu searches. At each iteration, a solution is derived from a randomized selection procedure which selects routes in the adaptive memory. In this section, we describe the procedure used to generate starting solutions. First, the routes in the adaptive memory are sorted in ascending order according to a weighted travel distance which is expressed as follows:

$$\hat{f}(r) = f(r) \cdot \frac{n_{\bar{r}}}{n} \tag{3}$$

r represents a route.
$f(r)$ is the distance travelled by r.
$n_{\bar{r}}$ is the number of customers not on r.
n is the total number of customers.
Let $l = (r_0, r_1, \ldots, r_{m-2}, r_{m-1})$ be the sorted list.
A non-negative score s_i is then associated with each route $r_i \in l$ as follows:

$$s_i = m - i \tag{4}$$

The procedure implemented for creating a new solution is probabilistically biased in favor of the best routes. We randomly generate a permutation of routes as follows:

$$p_i = \frac{s_i}{\sum_{k=0}^{k=m-1} s_k} \tag{5}$$

Where p_i is the probability of selecting the route r_i. Let P be a permutation of routes and S an empty solution. First, we insert the first route of P into S. Then, all routes of P that have one or more customers in common with this route are eliminated from the selection process. A second route is then selected among the remaining routes of P. This procedure is repeated until the selected routes serve all customers or until no admissible routes can be found. In the latter case, the insertion heuristic described in section 2.2.2 is used to insert the remaining customers. If this insertion procedure cannot serve all customers, additional routes are created for the remaining unserved customers.

3. Computational results

In this section, we compare the proposed cooperative parallel tabu search algorithm solution quality against published results. Our algorithm is implemented in Java. All experiments are carried out on an Intel(R) Core(TM) i7-6500U CPU 2.50GHz with 12 Go of RAM. We test our algorithm on the VRPTW benchmark instances. [7] created 81 benchmark instances for the VRPTW with 25, 50 and 100 customers distributed within $[0, 100]^2$ square. These instances are divided into three categories: C2, R2 and RC2. The customers are clustered in category C2, randomly distributed in category R2 and mixed in category RC2.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

These benchmark instances have large time windows; they are more representative of long-haul transportation with longer scheduling horizon and fewer vehicles. The travel distance between customers is equal to the corresponding Euclidean distance. To generate pseudo-random numbers, we use the combined multiple recursive pseudo-random number generator MRG32k3a [9]. MRG32k3a is a 32-bit combined multiple recursive generator with two components of order 3 that are expressed as follows:

$$y_{1,n} = (a_{11} \cdot y_{1,n-1} + a_{12} \cdot y_{1,n-2} + a_{13} \cdot y_{1,n-3}) \ mod \ m_1 \tag{6}$$

$$y_{2,n} = (a_{21} \cdot y_{2,n-1} + a_{22} \cdot y_{2,n-2} + a_{23} \cdot y_{2,n-3}) \ mod \ m_2 \tag{7}$$

The coefficients and modulo are defined as follows:
$a_{11} = 0$; $a_{12} = 1403580$; $a_{13} = -810728$; $m_1 = 2^{32} - 209$.
$a_{21} = 527612$; $a_{22} = 0$; $a_{23} = -1370589$; $m_2 = 2^{32} - 22853$.
MRG32k3a generator is defined as follows:

$$x_n = (y_{1,n} - y_{2,n}) \ mod \ m_1 \tag{8}$$

$$u_n = \frac{x_n}{m_1} \tag{9}$$

The sequence of integers $x_3, x_4, x_5, \ldots$ are the outputs of the generator. The pseudo-random numbers $u_3, u_4, u_5, \ldots$ are uniformly distributed. Note that MRG32k3a generator meets the requirements of modern generators such as good multidimensional uniformity and long period ($p \approx 2^{191}$). Before calling the procedure of generating pseudo-random numbers for the first time, one must initialize the global variables s_{10}, s_{11}, s_{12} to non-negative integer values less than m_1 and not all zero, and s_{20}, s_{21}, s_{22} to non-negative integers less than m_2 and not all zero. The vectors (s_{10}, s_{11}, s_{12}) and (s_{20}, s_{21}, s_{22}) are actually the initial values of $(y_{1,0}, y_{1,1}, y_{1,2})$ and $(y_{2,0}, y_{2,1}, y_{2,2})$ respectively. They constitute what we call the seed. In this computational study, we implement a diversification strategy that consists of using five different seeds for MRG32k3a generator. The used values are presented in Table 1. Besides, we suppose that $\alpha_1 \sim U[0,1]$, $\alpha_2 = \alpha_1 - 1$, μ and $\lambda \sim Exp(1)$.
On the other hand, preliminary experiments on a subset of VRPTW benchmark instances have been conducted to set parameter values of the proposed cooperative parallel tabu search algorithm. The following values are selected:

- $p = 10$; p denotes the number of slave threads.
- $I_{max}^{thread} = 50$; where I_{max}^{thread} denotes the maximum number of iterations allowed for every thread.
- $I_{NI_max}^{thread} = 10$; where I_{NI-max}^{thread} denotes the maximum number of consecutive iterations without an improvement for every thread.
- $M_{size} = 30$; where M_{size} denotes the size of the adaptive memory.
- $I_{max}^{tabu} = 500$; where I_{max}^{tabu} denotes the maximum number of iterations for every tabu search.
- $I_{NI_max}^{tabu} = 50$; where I_{NI-max}^{tabu} denotes the maximum number of iterations since the last improvement for every tabu search.
- $I_{tabu} = 15$; where I_{tabu} denotes the number of iterations after which an exchange marked as tabu is accepted.

The body of literature related to VRPTW and its variants is quite huge. Thus, trying to summarise such a huge literature is a challenging activity and a tiresome task. To systematize such a literature, one can present some recently published studies and the best performing approaches presented in literature to deal with VRPTW. [10] propose a column

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

generation based heuristic for the generalized vehicle routing problem with time windows. [11] propose a branch-price-and-cut algorithm for the two-echelon vehicle routing problem with time windows. [12] propose a variable neighborhood search for the two-echelon electric vehicle routing problem with time windows.

In this computational study, we restrict ourselves to studies that: deal with VRPTW as it was presented in this study without any additional constraint, present detailed computational results on VRPTW test instances so that we can compare with them and update best known solutions if they are any. Some of the best successful approaches proposed to deal with VRPTW include the following works:

C: the work by [13].
CC: the work by [14].
CR: the work by [15].
DLP: the work by [16].
IV: the work by [17].
JPSP: the work by [18].
KBM: the work by [19].
KLM: the work by [20].
L: the work by [21].
S: the work by [22].

We compare the performance of our cooperative parallel tabu search algorithm with that of all these algorithms which consist in exact and approximate algorithms. In the rest of this paper, we refer to our algorithm as CPTS algorithm for cooperative parallel tabu search algorithm.

We should notice that, computation times are not used for comparison due to possible variations in the configuration of hardware and software used. However, it is worth noting that our algorithm is relatively fast and most instances are solved in a short time.

Tables 2 to 6 present the obtained results. The column headings are defined as follows:

Instance-n: the name of the benchmark instance; the digits at the end give the number of customers.
NV: number of vehicles.
Distance: the total travelled distance.
BK: the optimal objective value when known or the best known solution value.
Authors: the work in which this result appears.

The results from Tables 2 to 6 show that: first of all, the efficiency of the proposed CPTS algorithm does not depend on the seed used. Indeed, efficient results are obtained whatever the seed used. On the benchmark instances with 25 and 50 customers, we often succeed in finding new best solutions. Indeed, our CPTS algorithm is able to achieve new best solutions visiting all customers using less vehicles compared to the best known solutions. For example on R201-50 instance, we are able to find a solution visiting all customers using only 3 vehicles; whereas the number of vehicles needed by the best known solution is 6. On the benchmark instances with 100 customers, very encouraging results are obtained. Indeed, we often succeed in finding a solution that uses the same number of vehicles to serve all customers as for the best known solution, but the travel distance is not as good as for the best known solution. That is the case for example of C204-100 instance where the total travel distance of CPTS solution and the best known solution is 645.9 and 588.1 respectively. On some instances, we are able to achieve a solution with a travel distance that is close to the distance travelled by the best known solution using the same number of vehicles. That is for example the case of C201-100 instance where the number of vehicles and travel distance of CPTS solution and the best known solution is 3 and 591.5, and, 3 and 589.1, respectively. In summary, the proposed CPTS algorithm is able to achieve high quality solutions on the VRPTW benchmark instances especially on instances with 25 and 50 customers. It is able to find 48 new best solutions which are given by Table 7.

 doi:10.20944/preprints202206.0074.v1

4. Conclusions

In this paper, a cooperative parallel tabu search meta-heuristic has been discussed for the vehicle routing problem with time windows. The solution method is based on the parallelization scheme in which several tabu search threads cooperate by asynchronously exchanging information on the best solutions founded so far. The exchanges are performed through a mechanism called adaptive memory which holds and manages a pool of solutions. This enforces the asynchronous strategy of information exchanges and ensures the independence of the individual search processes. We have succeeded in finding 48 new best solutions on the VRPTW benchmark instances. These new operational best solutions can be used for long-haul transportation with longer scheduling horizon and fewer vehicles. Given these results, it seems to us that the best thing to do now is trying to integrate the proposed solution method into a decision support system (DSS) as an optimization engine. We believe that it will be handy for freight dispatchers to benefit from the solution method embedded into it in their daily operational task of planning vehicle routes.

References

1. Laporte, G. What you should know about the vehicle routing problem. *Naval Research Logistics (NRL)* **2007**, *54*, 811–819.
2. Crainic, T.G.; Toulouse, M., Parallel Metaheuristics. In *Fleet Management and Logistics*; Crainic, T.G.; Laporte, G., Eds.; Springer US: Boston, MA, 1998; pp. 205–251.
3. Alba, E.; Luque, G.; Nesmachnow, S. Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research* **2013**, *20*, 1–48.
4. Crainic, T., Parallel Metaheuristics and Cooperative Search. In *Handbook of Metaheuristics*; Gendreau, M.; Potvin, J.Y., Eds.; Springer International Publishing: Cham, 2019; pp. 419–451.
5. Glover, F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* **1986**, *13*, 533–549. Applications of Integer Programming.
6. Badeau, P.; Guertin, F.; Gendreau, M.; Potvin, J.Y.; Taillard, E. A parallel tabu search heuristic for the vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies* **1997**, *5*, 109–122. Parallel Computing in Transport Research.
7. Solomon, M.M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research* **1987**, *35*, 254–265.
8. Taillard, E.; Badeau, P.; Gendreau, M.; Guertin, F.; Potvin, J.Y. A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows. *Transportation Science* **1997**, *31*, 170–186.
9. L'Ecuyer, P.; Meliani, L.; Vaucher, J. SSJ: a framework for stochastic simulation in Java. In Proceedings of the Proceedings of the Winter Simulation Conference, 2002, Vol. 1, pp. 234–242 vol.1.
10. Yuan, Y.; Cattaruzza, D.; Ogier, M.; Semet, F.; Vigo, D. A column generation based heuristic for the generalized vehicle routing problem with time windows. *Transportation Research Part E: Logistics and Transportation Review* **2021**, *152*, 102391.
11. Mhamedi, T.; Andersson, H.; Cherkesly, M.; Desaulniers, G. A Branch-Price-and-Cut Algorithm for the Two-Echelon Vehicle Routing Problem with Time Windows. *Transportation Science* **2022**, *56*, 245–264.
12. Akbay, M.A.; Kalayci, C.B.; Blum, C.; Polat, O. Variable Neighborhood Search for the Two-Echelon Electric Vehicle Routing Problem with Time Windows. *Applied Sciences* **2022**, *12*.
13. Chabrier, A. Vehicle Routing Problem with elementary shortest path based column generation. *Computers & Operations Research* **2006**, *33*, 2972–2990. Part Special Issue: Constraint Programming.
14. Czech, Z.; Czarnas, P. Parallel simulated annealing for the vehicle routing problem with time windows. In Proceedings of the Proceedings 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, 2002, pp. 376–383.
15. Cook, W.; Rich, J.L. A Parallel Cutting-Plane Algorithm for the Vehicle Routing Problem With Time Windows, 1999.
16. Danna, E.; Le Pape, C., Branch-and-Price Heuristics: A Case Study on the Vehicle Routing Problem with Time Windows. In *Column Generation*; Desaulniers, G.; Desrosiers, J.; Solomon, M.M., Eds.; Springer US: Boston, MA, 2005; pp. 99–129.
17. Irnich, S.; Villeneuve, D. The Shortest-Path Problem with Resource Constraints and k-Cycle Elimination for k $\geq$ 3. *INFORMS Journal on Computing* **2006**, *18*, 391–406.

 doi:10.20944/preprints202206.0074.v1

18. Jepsen, M.; Petersen, B.; Spoorendonk, S.; Pisinger, D. Subset-Row Inequalities Applied to the Vehicle-Routing Problem with Time Windows. *Operations Research* **2008**, *56*, 497–511.
19. Kallehauge, B.; Boland, N.; Madsen, O.B. Path inequalities for the vehicle routing problem with time windows. *Networks* **2007**, *49*, 273–293.
20. Kallehauge, B.; Larsen, J.; Madsen, O.B.G. Lagrangean Duality Applied on Vehicle Routing with Time Windows - Experimental Results. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2001.
21. Larsen, J. Parallelization of the Vehicle Routing Problem with Time Windows, 2001.
22. Salani, M. Branch-and-Price Algorithms for Vehicle Routing Problems. PhD thesis, 2006.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Table 1. Used seeds

Seed	s_{10}	s_{11}	s_{12}	s_{20}	s_{21}	s_{22}
1	12345	12345	12345	12345	12345	12345
2	-4083270913	2445121742	-1203640803	-2458677226	-1525149743	-2576425395
3	-2491146364	3759470475	-3700274381	1503429779	4200933272	2606273735
4	-1128235220	-705664733	-4149471649	924645467	1942604259	1707545926
5	-246707794	3939031502	-3519246097	-2574696983	2878922468	3581807064

Table 2. Results on VRPTW benchmark instances (seed 1)

Instance-n	BK			CPTS		Instance-n	BK			CPTS	
	NV	Distance	Authors	NV	Distance		NV	Distance	Authors	NV	Distance
C201-25	2	214.7	CR + L	2	215.5	R207-25	3	361.6	KLM	1	432.3
C201-50	3	360.2	CR + L	2	444.9	R207-50	3	575.5	JPSP	2	612.5
C201-100	3	589.1	CR + KLM	3	591.5	R207-100	2	890.6	CC	3	956.4
C202-25	2	214.7	CR + L	1	225.9	R208-25	1	328.2	IV + C	1	338.1
C202-50	3	360.2	CR + KLM	2	403.8	R208-50	2	487.7	KBM	2	498.3
C202-100	3	589.1	CR + KLM	3	716.6	R208-100	2	726.8	CC	3	788.3
C203-25	2	214.7	CR + L	1	223.3	R209-25	2	370.7	KLM	1	456.6
C203-50	3	359.8	CR + KLM	2	409.2	R209-50	4	600.6	IV + C	2	726.9
C203-100	3	588.7	KLM	3	687.9	R209-100	3	909.2	CC	3	1277.3
C204-25	2	213.1	CR + KLM	1	216.2	R210-25	3	404.6	CR + KLM	2	412.5
C204-50	2	350.1	KLM	2	367.7	R210-50	4	645.6	IV + C	2	716.4
C204-100	3	588.1	IV	4	689.8	R210-100	3	939.4	CC	3	1520.2
C205-25	2	214.7	CR + L	1	299.7	R211-25	2	350.9	KLM	1	371.4
C205-50	3	359.8	CR + KLM	2	439.9	R211-50	3	535.5	IV + DLP	2	594.2
C205-100	3	586.4	CR + KLM	3	629.9	R211-100	2	885.7	CC	3	915.5
C206-25	2	214.7	CR + L	1	285.4	RC201-25	3	360.2	CR + L	2	437.9
C206-50	3	359.8	CR + KLM	2	426.4	RC201-50	5	684.8	L + KLM	3	861.5
C206-100	3	586.0	CR + KLM	3	631.8	RC201-100	4	1406.9	CC	4	2300.9
C207-25	2	214.5	CR + L	1	274.9	RC202-25	3	338.0	CR + KLM	2	376.1
C207-50	3	359.6	CR + KLM	2	426.2	RC202-50	5	613.6	IV + C	3	721.6
C207-100	3	585.8	CR + KLM	3	595.3	RC202-100	3	1365.6	CC	4	1857.4
C208-25	2	214.5	CR + L	1	262.1	RC203-25	3	326.9	IV + C	1	437.6
C208-50	2	350.5	CR + KLM	2	352.1	RC203-50	4	555.3	IV + C	2	749.9
C208-100	3	585.8	KLM	3	599.4	RC203-100	3	1049.6	CC	4	1206.4
R201-25	4	463.3	CR + KLM	2	525.9	RC204-25	3	299.7	C	1	331.3
R201-50	6	791.9	CR + KLM	3	887.8	RC204-50	3	444.2	DLP	2	493.5
R201-100	4	1252.4	CC	4	1783.5	RC204-100	3	798.5	CC	3	971.6
R202-25	4	410.5	CR + KLM	2	457.8	RC205-25	3	338.0	L + KLM	2	415.9
R202-50	5	698.5	CR + KLM	3	757.4	RC205-50	5	630.2	IV + C	3	912.6
R202-100	3	1191.7	CC	4	1481.1	RC205-100	4	1297.6	CC	5	1529.2
R203-25	3	391.4	CR + KLM	2	403.9	RC206-25	3	324.0	KLM	1	550.2
R203-50	5	605.3	IV + C	3	654.2	RC206-50	5	610.0	IV + C	2	759.1
R203-100	3	939.5	CC	3	1318.4	RC206-100	3	1146.3	CC	4	1307.6
R204-25	2	355.0	IV + C	1	356.4	RC207-25	3	298.3	KLM	2	308.6
R204-50	2	506.4	IV	2	530.1	RC207-50	4	558.6	C	2	761.1
R204-100	2	825.5	CC	3	902.3	RC207-100	3	1061.1	CC	4	1185.0
R205-25	3	393.0	CR + KLM	1	503.7	RC208-25	2	269.1	C	1	321.7
R205-50	4	690.1	IV + C	2	780.9	RC208-50	3	476.7	S	2	531.1
R205-100	3	994.4	CC	3	1403.5	RC208-100	3	828.1	CC	3	1064.3
R206-25	3	374.4	CR + KLM	1	449.3						
R206-50	4	632.4	IV + C	2	722.8						
R206-100	3	906.1	CC	3	1169.6						

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Table 3. Results on VRPTW benchmark instances (seed 2)

Instance-n	BK			CPTS		Instance-n	BK			CPTS	
	NV	Distance	Authors	NV	Distance		NV	Distance	Authors	NV	Distance
C201-25	2	214.7	CR + L	2	215.5	R207-25	3	361.6	KLM	1	399.7
C201-50	3	360.2	CR + L	2	444.9	R207-50	3	575.5	JPSP	2	625.9
C201-100	3	589.1	CR + KLM	3	591.5	R207-100	2	890.6	CC	3	997.5
C202-25	2	214.7	CR + L	1	232.8	R208-25	1	328.2	IV + C	1	340.5
C202-50	3	360.2	CR + KLM	2	428.6	R208-50	2	487.7	KBM	2	496.7
C202-100	3	589.1	CR + KLM	4	678.5	R208-100	2	726.7	CC	3	827.5
C203-25	2	214.7	CR + L	1	223.3	R209-25	2	370.7	KLM	1	492.2
C203-50	3	359.8	CR + KLM	2	420.8	R209-50	4	600.6	IV + C	2	719.9
C203-100	3	588.7	KLM	3	759.1	R209-100	3	909.2	CC	3	1369.3
C204-25	2	213.1	CR + KLM	1	216.2	R210-25	3	404.6	CR + KLM	2	415.4
C204-50	2	350.1	KLM	2	373.4	R210-50	4	645.6	IV + C	2	864.4
C204-100	3	588.1	IV	4	683.9	R210-100	3	939.3	CC	4	1126.5
C205-25	2	214.7	CR + L	1	321.8	R211-25	2	350.9	KLM	1	366.6
C205-50	3	359.8	CR + KLM	2	439.9	R211-50	3	535.5	IV + DLP	2	589.6
C205-100	3	586.4	CR + KLM	3	672.7	R211-100	2	885.7	CC	3	933.5
C206-25	2	214.7	CR + L	1	285.4	RC201-25	3	360.2	CR + L	2	432.3
C206-50	3	359.8	CR + KLM	2	409.6	RC201-50	5	684.8	L + KLM	3	864.2
C206-100	3	586.0	CR + KLM	3	593.8	RC201-100	4	1406.7	CC	4	2044.6
C207-25	2	214.5	CR + L	1	277.3	RC202-25	3	338.0	CR + KLM	2	376.1
C207-50	3	359.6	CR + KLM	2	426.2	RC202-50	5	613.6	IV + C	3	737.5
C207-100	3	585.8	CR + KLM	3	588.3	RC202-100	3	1365.6	CC	4	1749.1
C208-25	2	214.5	CR + L	1	263.0	RC203-25	3	326.9	IV + C	1	438.3
C208-50	2	350.5	CR + KLM	2	352.3	RC203-50	4	555.3	IV + C	3	595.4
C208-100	3	585.8	KLM	3	591.0	RC203-100	3	1049.6	CC	4	1331.6
R201-25	4	463.3	CR + KLM	2	526.9	RC204-25	3	299.7	C	1	334.9
R201-50	6	791.9	CR + KLM	3	890.4	RC204-50	3	444.2	DLP	2	496.4
R201-100	4	1252.4	CC	4	1669.1	RC204-100	3	798.4	CC	3	1206.3
R202-25	4	410.5	CR + KLM	2	457.6	RC205-25	3	338.0	L + KLM	2	413.6
R202-50	5	698.5	CR + KLM	3	783.6	RC205-50	5	630.2	IV + C	3	990.3
R202-100	3	1191.7	CC	4	1388.2	RC205-100	4	1297.2	CC	4	1905.9
R203-25	3	391.4	CR + KLM	2	400.4	RC206-25	3	324.0	KLM	1	502.4
R203-50	5	605.3	IV + C	2	941.1	RC206-50	5	610.0	IV + C	2	762.5
R203-100	3	939.5	CC	4	1079.1	RC206-100	3	1146.3	CC	4	1334.9
R204-25	2	355.0	IV + C	1	405.7	RC207-25	3	298.3	KLM	2	308.6
R204-50	2	506.4	IV	2	516.8	RC207-50	4	558.6	C	2	722.7
R204-100	2	825.5	CC	3	874.4	RC207-100	3	1061.1	CC	4	1273.9
R205-25	3	393.0	CR + KLM	1	503.7	RC208-25	2	269.1	C	1	313.4
R205-50	4	690.1	IV + C	2	785.5	RC208-50	3	476.7	S	2	527.5
R205-100	3	994.4	CC	4	1196.9	RC208-100	3	828.1	CC	3	1077.6
R206-25	3	374.4	CR + KLM	1	487.1						
R206-50	4	632.4	IV + C	2	687.4						
R206-100	3	906.1	CC	3	1123.0						

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Table 4. Results on VRPTW benchmark instances (seed 3)

Instance-n	BK			CPTS		Instance-n	BK			CPTS	
	NV	Distance	Authors	NV	Distance		NV	Distance	Authors	NV	Distance
C201-25	2	214.7	CR + L	2	215.5	R207-25	3	361.6	KLM	1	426.6
C201-50	3	360.2	CR + L	2	444.9	R207-50	3	575.5	JPSP	2	665.8
C201-100	3	589.1	CR + KLM	3	591.5	R207-100	2	890.6	CC	3	966.3
C202-25	2	214.7	CR + L	1	223.3	R208-25	1	328.2	IV + C	1	329.3
C202-50	3	360.2	CR + KLM	2	404.2	R208-50	2	487.7	KBM	2	496.3
C202-100	3	589.1	CR + KLM	3	601.3	R208-100	2	726.7	CC	3	789.0
C203-25	2	214.7	CR + L	1	223.3	R209-25	2	370.7	KLM	1	439.6
C203-50	3	359.8	CR + KLM	2	424.6	R209-50	4	600.6	IV + C	2	751.2
C203-100	3	588.7	KLM	3	641.2	R209-100	3	909.2	CC	3	1201.4
C204-25	2	213.1	CR + KLM	1	218.2	R210-25	3	404.6	CR + KLM	2	425.4
C204-50	2	350.1	KLM	2	399.3	R210-50	4	645.6	IV + C	2	1052.9
C204-100	3	588.1	IV	3	657.3	R210-100	3	939.3	CC	3	1406.9
C205-25	2	214.7	CR + L	1	321.8	R211-25	2	350.9	KLM	1	366.6
C205-50	3	359.8	CR + KLM	2	448.9	R211-50	3	535.5	IV + DLP	2	588.5
C205-100	3	586.4	CR + KLM	3	610.7	R211-100	2	885.7	CC	3	938.5
C206-25	2	214.7	CR + L	1	285.4	RC201-25	3	360.2	CR + L	2	432.3
C206-50	3	359.8	CR + KLM	2	426.4	RC201-50	5	684.8	L + KLM	3	850.5
C206-100	3	586.0	CR + KLM	3	590.6	RC201-100	4	1406.7	CC	5	1637.3
C207-25	2	214.5	CR + L	1	274.8	RC202-25	3	338.0	CR + KLM	2	376.1
C207-50	3	359.6	CR + KLM	2	449.2	RC202-50	5	613.6	IV + C	3	700.3
C207-100	3	585.8	CR + KLM	3	798.8	RC202-100	3	1365.6	CC	4	1677.4
C208-25	2	214.5	CR + L	1	272.4	RC203-25	3	326.9	IV + C	1	434.9
C208-50	2	350.5	CR + KLM	2	352.1	RC203-50	4	555.3	IV + C	2	717.1
C208-100	3	585.8	KLM	3	588.3	RC203-100	3	1049.6	CC	4	1235.1
R201-25	4	463.3	CR + KLM	2	523.6	RC204-25	3	299.7	C	1	356.8
R201-50	6	791.9	CR + KLM	3	899.4	RC204-50	3	444.2	DLP	2	514.2
R201-100	4	1252.4	CC	4	1540.2	RC204-100	3	798.4	CC	3	1213.4
R202-25	4	410.5	CR + KLM	2	457.6	RC205-25	3	338.0	L + KLM	2	415.9
R202-50	5	698.5	CR + KLM	3	781.1	RC205-50	5	630.2	IV + C	3	819.2
R202-100	3	1191.7	CC	4	1356.2	RC205-100	4	1297.2	CC	4	1683.9
R203-25	3	391.4	CR + KLM	2	400.4	RC206-25	3	324.0	KLM	2	344.9
R203-50	5	605.3	IV + C	2	709.5	RC206-50	5	610.0	IV + C	2	772.3
R203-100	3	939.5	CC	4	1135.9	RC206-100	3	1146.3	CC	4	1341.0
R204-25	2	355.0	IV + C	1	396.7	RC207-25	3	298.3	KLM	2	308.6
R204-50	2	506.4	IV	2	523.6	RC207-50	4	558.6	C	2	698.3
R204-100	2	825.5	CC	3	899.5	RC207-100	3	1061.1	CC	4	1237.7
R205-25	3	393.0	CR + KLM	1	503.7	RC208-25	2	269.1	C	1	320.0
R205-50	4	690.1	IV + C	2	799.3	RC208-50	3	476.7	S	2	553.0
R205-100	3	994.4	CC	3	1286.7	RC208-100	3	828.1	CC	3	1040.9
R206-25	3	374.4	CR + KLM	1	413.2						
R206-50	4	632.4	IV + C	2	688.2						
R206-100	3	906.1	CC	3	1183.2						

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Table 5. Results on VRPTW benchmark instances (seed 4)

Instance-n	BK			CPTS		Instance-n	BK			CPTS	
	NV	Distance	Authors	NV	Distance		NV	Distance	Authors	NV	Distance
C201-25	2	214.7	CR + L	2	215.5	R207-25	3	361.6	KLM	1	401.7
C201-50	3	360.2	CR + L	2	444.9	R207-50	3	575.5	JPSP	2	621.0
C201-100	3	589.1	CR + KLM	3	591.5	R207-100	2	890.6	CC	3	1052.6
C202-25	2	214.7	CR + L	1	223.3	R208-25	1	328.2	IV + C	1	331.3
C202-50	3	360.2	CR + KLM	2	413.6	R208-50	2	487.7	KBM	2	493.7
C202-100	3	589.1	CR + KLM	3	714.9	R208-100	2	726.7	CC	3	811.5
C203-25	2	214.7	CR + L	1	223.3	R209-25	2	370.7	KLM	1	487.1
C203-50	3	359.8	CR + KLM	2	416.1	R209-50	4	600.6	IV + C	2	759.5
C203-100	3	588.7	KLM	3	632.7	R209-100	3	909.2	CC	3	1258.5
C204-25	2	213.1	CR + KLM	1	213.9	R210-25	3	404.6	CR + KLM	2	414.2
C204-50	2	350.1	KLM	2	360.4	R210-50	4	645.6	IV + C	2	732.3
C204-100	3	588.1	IV	3	645.9	R210-100	3	939.3	CC	4	1128.2
C205-25	2	214.7	CR + L	1	298.8	R211-25	2	350.9	KLM	1	377.3
C205-50	3	359.8	CR + KLM	2	444.1	R211-50	3	535.5	IV + DLP	2	582.4
C205-100	3	586.4	CR + KLM	3	649.6	R211-100	2	885.7	CC	3	979.8
C206-25	2	214.7	CR + L	1	285.4	RC201-25	3	360.2	CR + L	2	447.2
C206-50	3	359.8	CR + KLM	2	426.4	RC201-50	5	684.8	L + KLM	3	855.8
C206-100	3	586.0	CR + KLM	3	607.1	RC201-100	4	1406.7	CC	5	1553.4
C207-25	2	214.5	CR + L	1	274.8	RC202-25	3	338.0	CR + KLM	2	376.1
C207-50	3	359.6	CR + KLM	2	429.9	RC202-50	5	613.6	IV + C	3	702.8
C207-100	3	585.8	CR + KLM	3	606.7	RC202-100	3	1365.6	CC	4	1577.9
C208-25	2	214.5	CR + L	1	263.0	RC203-25	3	326.9	IV + C	1	435.9
C208-50	2	350.5	CR + KLM	2	352.1	RC203-50	4	555.3	IV + C	2	907.2
C208-100	3	585.8	KLM	3	588.5	RC203-100	3	1049.6	CC	3	1558.6
R201-25	4	463.3	CR + KLM	2	528.9	RC204-25	3	299.7	C	1	327.3
R201-50	6	791.9	CR + KLM	3	921.5	RC204-50	3	444.2	DLP	2	482.9
R201-100	4	1252.4	CC	4	1529.2	RC204-100	3	798.4	CC	3	1007.8
R202-25	4	410.5	CR + KLM	2	458.1	RC205-25	3	338.0	L + KLM	2	415.9
R202-50	5	698.5	CR + KLM	3	769.5	RC205-50	5	630.2	IV + C	3	799.1
R202-100	3	1191.7	CC	4	1386.4	RC205-100	4	1297.2	CC	4	1829.4
R203-25	3	391.4	CR + KLM	2	400.4	RC206-25	3	324.0	KLM	2	344.9
R203-50	5	605.3	IV + C	2	959.8	RC206-50	5	610.0	IV + C	2	774.7
R203-100	3	939.5	CC	4	1186.8	RC206-100	3	1146.3	CC	4	1307.1
R204-25	2	355.0	IV + C	1	434.6	RC207-25	3	298.3	KLM	2	308.6
R204-50	2	506.4	IV	2	515.8	RC207-50	4	558.6	C	2	723.7
R204-100	2	825.5	CC	3	948.1	RC207-100	3	1061.1	CC	4	1243.1
R205-25	3	393.0	CR + KLM	1	503.7	RC208-25	2	269.1	C	1	313.4
R205-50	4	690.1	IV + C	2	814.7	RC208-50	3	476.7	S	2	563.4
R205-100	3	994.4	CC	3	1305.9	RC208-100	3	828.1	CC	3	1054.8
R206-25	3	374.4	CR + KLM	1	431.1						
R206-50	4	632.4	IV + C	2	744.6						
R206-100	3	906.1	CC	3	1165.8						

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 6 June 2022 doi:10.20944/preprints202206.0074.v1

Table 6. Results on VRPTW benchmark instances (seed 5)

Instance-n	BK			CPTS		Instance-n	BK			CPTS	
	NV	Distance	Authors	NV	Distance		NV	Distance	Authors	NV	Distance
C201-25	2	214.7	CR + L	2	215.5	R207-25	3	361.6	KLM	1	401.7
C201-50	3	360.2	CR + L	2	444.9	R207-50	3	575.5	JPSP	2	625.1
C201-100	3	589.1	CR + KLM	3	591.5	R207-100	2	890.6	CC	3	1142.1
C202-25	2	214.7	CR + L	1	223.3	R208-25	1	328.2	IV + C	1	331.3
C202-50	3	360.2	CR + KLM	2	443.9	R208-50	2	487.7	KBM	2	503.8
C202-100	3	589.1	CR + KLM	4	695.7	R208-100	2	726.7	CC	3	805.0
C203-25	2	214.7	CR + L	1	223.3	R209-25	2	370.7	KLM	1	451.5
C203-50	3	359.8	CR + KLM	2	424.1	R209-50	4	600.6	IV + C	2	781.1
C203-100	3	588.7	KLM	4	679.9	R209-100	3	909.2	CC	3	1464.3
C204-25	2	213.1	CR + KLM	1	215.2	R210-25	3	404.6	CR + KLM	2	410.6
C204-50	2	350.1	KLM	2	359.9	R210-50	4	645.6	IV + C	2	758.4
C204-100	3	588.1	IV	3	690.8	R210-100	3	939.3	CC	4	1200.9
C205-25	2	214.7	CR + L	1	297.4	R211-25	2	350.9	KLM	2	386.0
C205-50	3	359.8	CR + KLM	2	439.9	R211-50	3	535.5	IV + DLP	2	610.2
C205-100	3	586.4	CR + KLM	3	615.7	R211-100	2	885.7	CC	3	949.3
C206-25	2	214.7	CR + L	1	285.4	RC201-25	3	360.2	CR + L	2	432.3
C206-50	3	359.8	CR + KLM	2	409.9	RC201-50	5	684.8	L + KLM	3	861.5
C206-100	3	586.0	CR + KLM	3	588.5	RC201-100	4	1406.7	CC	5	1719.7
C207-25	2	214.5	CR + L	1	274.8	RC202-25	3	338.0	CR + KLM	2	376.9
C207-50	3	359.6	CR + KLM	2	426.2	RC202-50	5	613.6	IV + C	3	793.5
C207-100	3	585.8	CR + KLM	3	606.7	RC202-100	3	1365.6	CC	4	1651.8
C208-25	2	214.5	CR + L	1	234.9	RC203-25	3	326.9	IV + C	1	434.1
C208-50	2	350.5	CR + KLM	2	352.3	RC203-50	4	555.3	IV + C	2	805.8
C208-100	3	585.8	KLM	3	590.8	RC203-100	3	1049.6	CC	4	1212.1
R201-25	4	463.3	CR + KLM	2	523.7	RC204-25	3	299.7	C	1	356.8
R201-50	6	791.9	CR + KLM	3	903.4	RC204-50	3	444.2	DLP	2	499.7
R201-100	4	1252.4	CC	5	1382.4	RC204-100	3	798.4	CC	3	1212.6
R202-25	4	410.5	CR + KLM	2	458.1	RC205-25	3	338.0	L + KLM	2	415.9
R202-50	5	698.5	CR + KLM	3	773.1	RC205-50	5	630.2	IV + C	3	787.8
R202-100	3	1191.7	CC	4	1443.1	RC205-100	4	1297.2	CC	4	1905.1
R203-25	3	391.4	CR + KLM	2	400.4	RC206-25	3	324.0	KLM	2	345.3
R203-50	5	605.3	IV + C	2	737.7	RC206-50	5	610.0	IV + C	2	845.7
R203-100	3	939.5	CC	4	1137.5	RC206-100	3	1146.3	CC	4	1328.6
R204-25	2	355.0	IV + C	2	356.6	RC207-25	3	298.3	KLM	2	309.1
R204-50	2	506.4	IV	2	516.4	RC207-50	4	558.6	C	2	710.2
R204-100	2	825.5	CC	3	910.1	RC207-100	3	1061.1	CC	4	1352.7
R205-25	3	393.0	CR + KLM	1	503.7	RC208-25	2	269.1	C	1	310.5
R205-50	4	690.1	IV + C	2	797.8	RC208-50	3	476.7	S	2	591.9
R205-100	3	994.4	CC	4	1132.1	RC208-100	3	828.1	CC	3	1165.4
R206-25	3	374.4	CR + KLM	1	437.1						
R206-50	4	632.4	IV + C	2	745.9						
R206-100	3	906.1	CC	3	1288.6						

Table 7. New best solutions found by CPTS algorithm

Instance-n	NV	Distance	Instance-n	NV	Distance	Instance-n	NV	Distance
C202-25	1	223.3	R211-25	1	366.6	R202-50	3	757.4
C203-25	1	223.3	RC201-25	2	432.3	R203-50	2	709.5
C204-25	1	213.9	RC202-25	2	376.1	R205-50	2	780.9
C205-25	1	297.4	RC203-25	1	434.1	R206-50	2	687.4
C206-25	1	285.4	RC204-25	1	327.3	R207-50	2	612.5
C207-25	1	274.8	RC205-25	2	413.6	R209-50	2	719.9
C208-25	1	234.9	RC206-25	1	502.4	R210-50	2	716.4
R201-25	2	523.6	RC207-25	2	308.6	R211-50	2	582.4
R202-25	2	457.6	RC208-25	1	310.5	RC201-50	3	850.5
R203-25	2	400.4	C201-50	2	444.9	RC202-50	3	700.3
R204-25	1	356.4	C202-50	2	403.8	RC203-50	2	717.1
R205-25	1	503.7	C203-50	2	409.2	RC204-50	2	482.9
R206-25	1	413.2	C205-50	2	439.9	RC205-50	3	787.7
R207-25	1	399.7	C206-50	2	409.6	RC206-50	2	759.1
R209-25	1	439.6	C207-50	2	426.2	RC207-50	2	698.3
R210-25	2	410.6	R201-50	3	887.8	RC208-50	2	527.5