ISSN 2007-9737

A Cooperative Algorithm to Tackle the Truck and Trailer Routing Problem

Ricardo Pérez-Rodríguez[1], Luis Eduardo Urbán-Rivero[2,3,*]

[1] CONAHCYT,
Mexico

[2] Instituto Tecnológico Autónomo de México,
Mexico

dr.ricardo.perez.rodriguez@gmail.com, luis.urban@itam.mx

Abstract. In this paper, we proposed a cooperative algorithm to resolve the truck and trailer routing problem. In this proposal, each member of the population does not represent a complete solution as in almost any evolutionary algorithm. In addition, for each member, an aptitude is not possible to compute based only on its codification, because the member has only partial information of the solution. All the members of the population have partial information of the solution. Therefore, these members need to cooperate to obtain an aptitude for the entire population. Although diverse methods and strategies have been used to solve the TTRP, this paper contributes to the state of the art by the use of the idea of the absence of recombination. We adopt the idea of no communication among members of the population. This way of computing fitness is clearly a gap in the literature, and must be investigated. Enough experimental results are shown that the cooperative algorithm is competitive against other current evolutionary algorithms. There no exist statistically significant difference between the cooperative algorithm and the others. It means that the CoopA is a new approach to continue developing.

Keywords. Cooperative algorithm, evolutionary algorithms, vehicle routing problems, set-partitioning model.

1 Introduction and Related Work

sec01) Since some decades ago, the performance of evolutionary computation has significantly improved, enabling the resolution of various optimization problems. From basic procedures to very elaborate hybrid methods, all of them efficiently address most NP-hard issues. Normally, any optimization algorithm, specifically evolutionary algorithms, involves creating a population of solutions, selecting certain members based on their aptitude, reproducing them to generate offspring, and repeating this process until the best solution is obtained. We are used to developing optimization algorithms using the same procedures, such as initial population, selection, crossover, mutation, and replacement. This is because we perceive the survival of the species as a well-defined process. Although almost all optimization algorithms share this feature, there are other algorithms inspired by different natural processes, such as particle swarm optimization and artificial immune systems. In fact, some evolutionary algorithms, particularly 'evolutionary programming' and older varieties of evolution strategy, do not use recombination at all [13]. Therefore, there are alternative possibilities for creating optimization algorithms that are not inspired by the aforementioned population-based procedures of evolutionary algorithms.

An example is asexual living beings. Asexual reproduction enables living beings to transmit their genetic information to their descendants without the union of information (gametes) from individuals. In real life, jellyfish, corals, and sea sponges are examples of the absence of recombination.

There are already algorithms composed of various populations that do not communicate with each other but rather evolve independently. The research by Absi et al [1] falls into this category.

In this paper, we propose the absence of recombination and the idea of no communication among members of the population.

In this research, the concept of competition among members of the population is not considered. On the contrary, members of the population participate and cooperate to build a global aptitude for the entire population. The global aptitude is obtained through the contribution of all members, without individual fitness values. Members are not selected for specific purposes, and offspring can be generated through either classical procedures or alternative methods, depending on each researcher's approach. The replacement process between offspring and parents is decisive, with the population exhibiting the best fitness surviving and completely replacing the less fit population. This process can be likened to a binary tournament between parents and offspring. Given these characteristics, we refer to this algorithm as the Cooperative Algorithm (CoopA), utilized to solve the truck and trailer routing problem (TTRP), which is an NP-hard issue.

Normally, in almost any evolutionary algorithm, each member of the population represents a solution, and aptitude is computed based on its codification. However, in CoopA, it is not possible to compute aptitude based solely on a member's codification because each member has only partial information about the solution. All members of the population possess partial information. Therefore, cooperation among members is essential to obtain an aptitude for the entire population. This method of computing fitness represents a gap in the literature and warrants further investigation.

The Truck and Trailer Routing Problem (TTRP) is particularly suited for resolution using the novel CoopA scheme. In essence, the TTRP involves delivering products to customers using a combination of trucks and trailers. Vehicles depart from a central depot to traverse various routes, with each vehicle returning to the depot upon completing its journey.

Beyond capacity constraints, the TTRP incorporates operational restrictions, such as limited trailer access at certain customer locations. This limitation arises due to factors like narrow spaces for maneuvers, traffic restrictions, and other considerations. Consequently, vehicles must be parked elsewhere, trailers unhitched, and the journey continued using only the truck before reaching customers with restricted trailer access. After product delivery, the truck returns to retrieve the trailer before continuing its journey.

The unique considerations outlined above give rise to three main types of routes. The first type exclusively employs trucks for product delivery, termed 'truck routes.' The second type, called 'vehicle routes,' involves using both the truck and trailer for customers with unrestricted access. The third type, 'mix routes,' also employs both the truck and trailer but requires unhitching when encountering customers with maneuvering restrictions.

Customers who only permit access to trucks at their facilities are labeled 'truck customers,' while those allowing access to both trucks and trailers are termed 'vehicle customers.' In the TTRP, it is possible to park and unhitch the trailer at any vehicle customer location before delivering products to truck customers on the trip, see Fig 1.

Finally, the main objective is to select a set of routes, minimizing the total distance traveled, to efficiently deliver products to all customers."

Based on the characteristics of the TTRP, it is suitable to apply the CoopA. The purpose is to build many routes as possible, and all of them are built through the information of the members of the population. Each member participates and cooperates with a set of routes in order to find and select the best of them, i.e., of minimum total distance. Thus, each member has only partial information of the solution, i.e., a set of routes. Therefore, the fitness of the population is obtained by choosing from the routes of minimum total distance.

Currently the majority of methods to resolve vehicle routing problems use a conventional mechanism to build solutions, i.e., group customers in a route, and then sequence the route. It is commonly named 'cluster-first route-second'. The constraints of the problem being analyzed are considered to group customers.

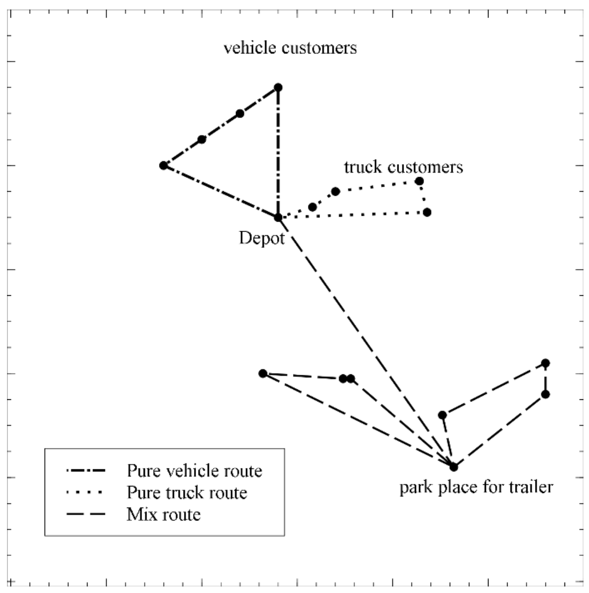

Fig. 1. Sort of routes in the TTRP

Prins et al [15] cited contributions of this approach for vehicle routing optimization problems, such as [19, 8, 23] cited important contributions for the TTRP using this approach. Examples of applications in real-world situations are found in [5, 7, 9, 2, 20, 22, 28]. However, for two decades, an alternative approach has had increasing acceptance, i.e., the 'route-first cluster-second' mechanism. This relatively new approach has led to successful methods for routing problems. It is due to its flexibility and efficiency. Such properties have let to resolve the TTRP too. Again in [15] and [23] cited the most relevant papers in this category, such as [3, 10, 14, 16, 17, 24, 25, 23]

In the route-first cluster-second, generally each solution for the TTRP, is represented for a permutation of vertices. Therefore, after the split-phase, a fitness is obtained for that solution. In this research, also we adopt the route-first cluster-second approach, but we differ in the split-phase to build routes. This dissimilarity permits to represent of a member of the CoopA population as a route. Thus, each route is a member of the population in the core of the CoopA. It is then clear that each route only contains partial information of the solution and it is necessary to consider all the routes to generate a population fitness.

Basically, the TTRP's current research is focused on the use of heuristics and metaheuristics. The tabu-search algorithm is used to tackle the TTRP in [4]. With tabu search, the author allocates customers to routes at the beginning, followed by an insertion heuristic. Scheuerer employs two heuristics in [18] to develop initial solutions, and later the solutions are improved through tabu search. In [18] the authors address the TTRP through sequential heuristics. First, assigns customers to valid routes, and then defines the sequence of each route In [18]. Yu et al tackle the TTRP by an ant colony system to build feasible solutions, and then these solutions are improved by a process improvement for each solution.

In [11], Lin et al detail a heuristic based on SA technique for the TTRP. In [28] Authors extends the idea to address the time window constraints.

Villegas et al in [23] detail a hybrid Greedy Randomized Adaptive Search Procedure (GRASP) with Variable Neighbourhood Search (VNS) heuristic for the TTRP. In [24] Villegas et al coupled this heuristic with a set-partitioning formulation to tackle the same problem.

If time windows for delivery exist, and the option of load transfer between truck and trailer is required, the paper of Derigs et al [6] is suitable when we need to analyze the Rich Vehicle Routing Problem (RVRP). The study details a flexible hybrid approach, which is based on local search and large neighborhood.

In [1], Absi et al propose an evolutionary algorithm composed of multiple populations that evolve independently, without communication, to solve the TTRP (Truck and Trailer Routing Problem).

In [12] Maghfiroh and Hanaoka solve a dynamic truck and trailer routing problem for last mile distribution in disaster response by a modified simulated annealing algorithm with variable neighborhood search for local search. The fitness in this research is the total travel time. For dealing with the stochastic and dynamicity of the problem, a dynamic simulator is added to the framework to incorporate new requirements of the customers.

In [27] Wang et al detail a bat algorithm (BA) to tackle the TTRP. The procedure uses five different neighborhood structures as part of local search strategy. Moreover, to preserve diversity, a self-adaptive (SA) tuning strategy is used in the proposed algorithm.

In [29] Yuan et al tackle the TTRP by a Backtracking Search Algorithm (BSA). The algorithm uses four types of route improvement to produce offspring, and a T-sweep heuristic to build the initial population.

Although diverse methods and strategies have been used to solve the TTRP, this paper contributes to the state of the art by the use of the idea of absence of recombination. We adopt the idea of no communication among members of the population.

2 A Set-Partitioning Formulation for the TTRP

Previously we discuss the possible routes that can be built in the TTRP, i.e., truck routes, vehicle routes, and mix routes. Also, erstwhile we refer to truck customers, and vehicle customers. Then we can establish binary parameters for each sort of route. It means:

Parameters:

J : Set of feasible truck routes.

K : Set of feasible vehicle routes.

M : Set of feasible mix routes.

N_t: Set of truck customers.

N_t: Set of vehicle customers.

d_j represents the total distance of the truck route j.

d_k represents the total distance of the vehicle route k.

d_m represents the total distance of the mix route m.

Variables:

$$a_{i,j} = \begin{cases} 1 & \text{If the customer } i \text{ is visited by the truck route } j. \\ 0 & \text{Otherwise.} \end{cases}$$

$$b_{i,k} = \begin{cases} 1 & \text{If the customer } i \text{ is visited by the vehicle route } k. \\ 0 & \text{Otherwise.} \end{cases}$$

$$c_{i,m} = \begin{cases} 1 & \text{If the customer } i \text{ is visited by the mix route } m. \\ 0 & \text{Otherwise.} \end{cases}$$

$$x_j = \begin{cases} 1 & \text{If the truck route j is selected and used in the solution for TTRP.} \\ 0 & \text{Otherwise.} \end{cases}$$

$$y_k = \begin{cases} 1 & \text{If the vehicle route k is selected and used in the solution for TTRP.} \\ 0 & \text{Otherwise.} \end{cases}$$

$$z_m = \begin{cases} 1 & \text{If the mix route m is selected and used in the solution for TTRP.} \\ 0 & \text{Otherwise.} \end{cases}$$

$$\min \ z = \sum_{j \in J} d_j x_j + \sum_{k \in K} d_k y_k + \sum_{m \in M} d_m y_m, \quad (1)$$

$$\sum_{j \in J} a_{ij} x_j + \sum_{k \in K} b_{ik} y_k + \sum_{m \in M} c_{im} z_m = 1 \ \ \forall i \in N_v, \quad (2)$$

$$\sum_{j \in J} a_{ij} x_j + \sum_{m \in M} c_{im} z_m = 1 \ \ \forall i \in N_t. \quad (3)$$

The objective function (1) consists of the first part that corresponds to the total distance of truck routes, the second part represents the total distance of vehicle routes, and the third part is the total distance of mix routes. Constraints (2) assure that each vehicle customer is visited exactly once; whereas, constraints (3) assure that each truck customer is visited exactly once by a truck route or by a mix route.

3 CoopA Framework

3.1 Route-First Step

A permutation representation is built to execute the route-first step. The CoopA uses five different procedures to build permutation representations, i.e., trips. All the procedures are well-known techniques in the literature.

3.1.1 Random Insertion

The first one, it is a random procedure, where all the vertices are positioned on the trip randomly, i.e., the procedure randomly selects a vertex from a list of not visited vertices and inserts it in the trip. Update the list of not visited vertices, and repeat the procedure until all the vertices are included in the trip. The Algorithm 1 shows the random insertion procedure.

Algorithm 1 Random insertion

```
$V \leftarrow$ A list of non visited vertices
$T \leftarrow$ An empty trip
do
    $v_s \leftarrow$ Randomly select a vertex of $V$
    $T \leftarrow T \cup v_s$
    $V \leftarrow V \backslash v_s$
while $V \neq \emptyset$
return $T$
```

3.1.2 Nearest Neighbour Procedure

The second, the nearest neighbor procedure starts at one vertex (randomly selected from a list of not visited vertices and inserts it in the trip), update the list of not visited vertices, identifies the closest unvisited vertex, and inserts the closest unvisited vertex to the trip, again update the list of not visited vertices. It repeats until every vertex has been visited. Algorithm 2 shows the Nearest neighbour procedure.

Algorithm 2 Nearest neighbour

```
$V \leftarrow$ A list of non visited vertices
$T \leftarrow$ An empty trip
$v_s \leftarrow$ Randomly select a vertex of $V$
$T \leftarrow T \cup v_s$
$V \leftarrow V \backslash v_s$
do
    $v_n \leftarrow$ Select the closest non visited vertex to $v_s$
    $T \leftarrow T \cup v_n$
    $V \leftarrow V \backslash v_n$
    $v_s \leftarrow v_n$
while $V \neq \emptyset$
return $T$
```

3.1.3 Adaptative Nearest Neighbor Procedure

The third, the nearest neighbor procedure from both end-points, where it starts with a vertex chosen randomly. Then, it continues with the nearest unvisited vertex to this vertex. We will have two end vertices. We add a vertex to the trip such that this vertex has not visited before and it is the nearest vertex to these two end vertices. We update the end vertices. It ends after visiting all the vertices. Algorithm 3 shows the Adaptative nearest neighbour procedure.

3.1.4 Nearest Insertion Procedure

The fourth, the nearest insertion procedure, where it begins with two vertices. It then repeatedly finds the vertex not already in the trip that is closest to any vertex in the trip, and places it between whichever two vertices would cause the resulting trip to be the shortest possible. It stops when no more insertions remain. Algorithm 4 shows the Nearest insertion procedure.

3.1.5 2-Opt Procedure

Finally, the fifth, the 2-Opt procedure proposed by Croes (1958), where it originates from the idea that trips with edges that cross over are not optimal. 2-Opt will consider every possible 2-edge swap, swapping 2 edges when it results in an improved trip. Algorithm 5 shows the 2-Opt procedure.

Algorithm 3 Adaptative nearest neighbour

```
$V \leftarrow$ A list of non visited vertices
$T \leftarrow$ An empty trip
$v_a \leftarrow$ Randomly select a vertex of $V$
$v_b \leftarrow v_a,\ T \leftarrow T \cup v_a,\ \ V \leftarrow V \backslash v_a$
do
    if $|T|$ = 1 then
        $n_a \leftarrow$ Select the closest non visited
        vertex to v$_a$
        $v_a \leftarrow n_a,\ \ T \leftarrow T \cup n_a,\ \ V \leftarrow V \backslash n_a$
    else
        $n_a \leftarrow$ Select the closest non visited
        vertex to v$_a$
        $n_b \leftarrow$ Select the closest non visited
        to vertex to v$_b$
        if $n_a < n_b$ then
            $v_a \leftarrow n_a,\ \ T \leftarrow T \cup n_a,\ \ V \leftarrow V \backslash n_a$
        else
            $v_b \leftarrow n_b,\ T \leftarrow T \cup n_b,\ \ V \leftarrow V \backslash n_b$
        end if
    end if
while $V \neq \emptyset$
return $T$
```

3.2 Cluster-second step

Each trip, obtained in the previous step, is split as many feasible routes as possible. For that purpose, three variants are used to build feasible routes.

3.2.1 Building Feasible Truck Routes

In this case, we read each trip from left to right. Let a trip $T = \{v_i, \ldots, v_j, \ldots, v_k\}$, we confirm the expression $q_i \leq Q_t$, it means that if the demand of the vertex i is less or equal to the capacity truck Q_t, then the vertex i can belong to the route. Otherwise, the route is finished. The process continues reading the trip, and we update the total demand on this route if the next vertex j can be considered on the route, i.e., if the vertex j meets $Q_{i,j} \leq Q_t$ where $Q_{i,j} = \sum_{u=i}^{j} q_{v_u}$. We will stop when such condition is not met, then the route is finished. The process continues reading the rest of the trip, and it finishes when all the vertices have already been assigned to some route.

Algorithm 4 Nearest insertion procedure

```
$V \leftarrow$ A list of non visited vertices
$T \leftarrow$ An empty trip
$v_s \leftarrow$ Randomly select a vertex of $V$
$T \leftarrow T \cup v_s$
$V \leftarrow V \backslash v_s$
$v_n \leftarrow$ Randomly select a vertex of $V$
$T \leftarrow T \cup v_n$
$V \leftarrow V \backslash v_n$
do
    $v_a \leftarrow$ Identify the closest non visited vertex
        any vertex of $T$
    $p \leftarrow$ Identify where insert $v_a$ cause $T$ be
        the shortest
  $T \leftarrow$ Insert $v_a$ at the position $p$ of the trip $T$
  $V \leftarrow V \backslash v_a$
while $V \neq \emptyset$
return $T$
```

3.2.2 Building Feasible Vehicle Routes

The process is very similar than the previous one. The main difference is found in the capacity of the vehicle, which is no longer Q_t, will be $Q_t + Q_r$, i.e., the capacity of the truck plus the capacity of the trailer. In addition, we need to verify if the vertex i can receive a trailer. Otherwise, the route is finished. The process continues reading the trip, and we update the total demand on this route if the next vertex j meets the restriction of capacity and reception of a trailer. We will stop when such conditions are not met, then the route is finished. The process continues reading the rest of the trip, and it finishes when the vertices able to receive a trailer have already been assigned to some route.

3.2.3 Building Feasible Mix Routes

The process starts reading the trip as the previous ones. The capacity of the vehicle is $Q_t + Q_r$. We confirm the expression $q_i \leq Q_t + Q_r$, then the vertex i can belong to the route. Otherwise, the route is finished. The process continues reading the trip, and we update the total demand on this route if the next vertex j can be considered on the route, i.e., if the vertex j meets $Q_{ij} \leq Q_t +$

Algorithm 5 2-Opt procedure

```
$T \leftarrow$ Select randomly a trip
$Y^* \leftarrow$ Compute total distance of $T$
$Imp \leftarrow$ **true**
**while** $Imp =$ **true do**
    $Imp \leftarrow$ **false**
    **for** $i \leftarrow 1$ **to** $|V|$ **do**
        **for** $j \leftarrow i$ **to** $|V|$ **do**
            $T_a \leftarrow swap(T, i, j)$
            $Y_a \leftarrow$ Compute the total distance of $T_a$
            **if** $Y_a < Y^*$ **then**
                $T \leftarrow T_a$
                $Y^* \leftarrow Y_a$
                $Imp \leftarrow$ **true**
            **end if**
        **end for**
    **end for**
**end while**
**return** $T$
```

Q_r where $Q_{ij} \sum_{u=i}^{j} q_{v_u}$. We will stop when such condition is not met, then the route is finished.

The next step is to verify if the route, already built, contains at least one truck customer. If so then, we confirm that the first vertex on the route be a vehicle customer. If so then, the route is a mix route, and we park and unhitch the trailer is that first vertex. If not then, the route is unfeasible and it is discarded. The process continues reading the rest of the trip, and it finishes when we have already analyzed all the vertices on the trip.

All the routes built by these three variants are members of the population, in the CoopA framework. All the routes are considered to find a fitness for the population.

3.3 Total Distance Computing

For each route built by any of the three aforementioned variants, a total distance is computed. The total distance for the truck routes and the vehicle routes is easily computed because it corresponds to a single tour, without forgetting that the route leaves the depot and returns at the end. The total distance for the mix routes is computed considering that the trailer is unhitch at the first vehicle customer location on the route, after that the truck visits one or more customers on the route, probably the truck has to come back to the parking place of the trailer to transfer product between the trailer and the truck, and continue the tour until satisfying pending customers. We emphasize that the route leaves the depot, sometime the truck has to return to hitch the trailer, and finally the vehicle goes back to the depot at the end.

3.4 Fitness of the Population

The mathematical model, detailed in Section II, is applied to minimize the total distance of the solution, i.e., the fitness of the population. This model considers all the routes built in Section III-B, the total distance of each route computed in Section III-C., to identify the minimum, and know which routes are elected.

3.5 Offspring Population

Again, we create trips by five different procedures. Four of them, have been previously detailed in Section III-A, i.e., the nearest neighbor technique, the nearest neighbor technique from both end-points, the nearest insertion technique, and the 2-Opt technique.

The fifth procedure is the partially mapped crossover, called PMX genetic operator. Here, we select randomly two trips, obtained in Section III-A, and we apply the PMX operator to produce one new trip. The process detailed in III-B, is repeated to produce feasible routes that we consider as the offspring population in the CoopA framework. The processes III-C, and III-D, are repeated to know the fitness of the offspring population.

3.6 Replacement

Although the population with the best fitness survives, the best trips of both populations are preserved to build feasible routes in the next generation.

The parameters used are detailed below.

— 25 generations

— 50 trips per generation

The CoopA framework is provided in Algorithm 6

Algorithm 6 CoopA framework

```
    $D_0 \leftarrow$ Generate $M$ trips
    $R_0 \leftarrow$ Build feasible routes $f \in D_0$
    $Dist[R_0] \leftarrow$ Compute distance of each
                    route of $R_0$
$Y^* \leftarrow Fitness(R_0, Dist[R_0])$
$R^* \leftarrow$ Store the best feasible route
$Dist[R^*] \leftarrow$ Store the best distance of the best
            feasible routet $\leftarrow 1$
do
    $S_t \leftarrow$ Generate $M$ trips
    $R_t \leftarrow$ Build feasible routes from $S_t$
    $Dist[R_t] \leftarrow$ Compute distance
                    of each route of $R_t$
    $BestOffspring \leftarrow Fitness(R_t, Dist[R_t])$
    $Y^* \leftarrow$ If apply, update the best solution
            of BestOffspring
    $R^* \leftarrow$ If apply, replacement of $R^*$
    $Dist[R^*] \leftarrow$ If apply, replacement of $Dist[R^*]$
    $t \leftarrow t + 1$
while not Stop critterion meet
return $Y^*$
```

4 Results and Comparison

The CoopA is compared with other evolutionary algorithms in order to show its performance. The comparison is done using the algorithm detailed by Derigs et al in [6], the simulated annealing heuristic designed by Maghfiroh and Hanaoka in [12], and the bat algorithm presented by Wang et al [27]. All these algorithms were implemented following the available information. The proposed approach is applied to Chao's 21 TTRP benchmark problems in [4] available on the web http://140.118.201.170/ttrp/

The Chaos´s benchmark problems include

— On the first line of each instance, the capacity of the trucks, the capacity of the trailers, and the available number of the vehicles.

— From the second line of each instance to the end, the information of each customer, i.e., the number of customer and depot, the X coordinate, the Y coordinate, the demand, and availability to accept the complete vehicle in its location.

Fig. 2 details the performance for each algorithm.

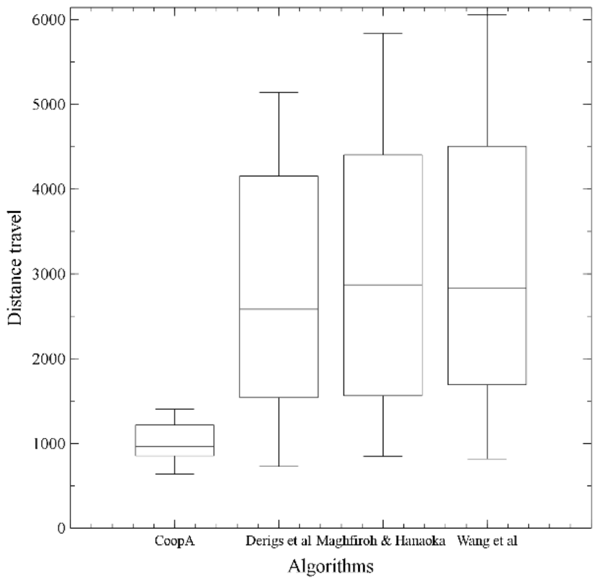

Fig. 2. First computational results

Based on Fig. 2, the dispersion of the results is less in the CoopA than others. It is due to the replacement procedure, detailed in section 3-F., keeps the best fitness over all the iterations, and the average of each generation cannot be far away from the best solution because the most offspring are built by the same procedures than the parents.

In addition, another comparison is presented in Fig. 3. It is using the algorithm proposed by Lin et al. in [11], and the procedure shown by Villegas et al. in [24] for comparison with the CoopA scheme. The results of these algorithms were taken directly from the available literature, and the same dataset (Chao's 21 TTRP benchmark problems) was used in this comparative.

Based on Fig. 3, the dispersion of the results is very similar among the algorithms. The performance of CoopA is competitive. It is due to the large number of routes built in each instance. We devised procedures, detailed in section III-B., to tackle the most drawback of the set-partitioning model for the TTRP, i.e., the structure of mix routes that normally are resolved by column generation and branch-and-price methods (Drexl,

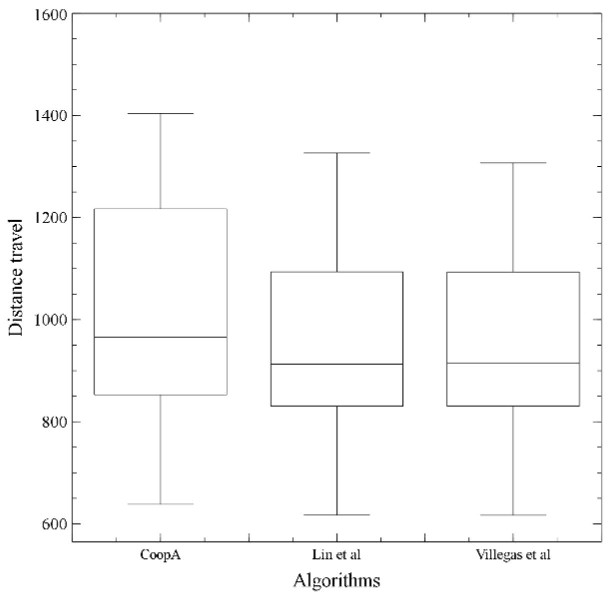

Fig. 3. Second computational results

2011). Furthermore, in this research, we do not use any auxiliary graph to build feasible routes.

A Dunnett test [21] is done to identify if there exist statistically significant difference between CoopA and the other methods. The CoopA is competitive, there no exist statistically significant difference (see Fig. 4). It means that CoopA is a new approach to continue developing.

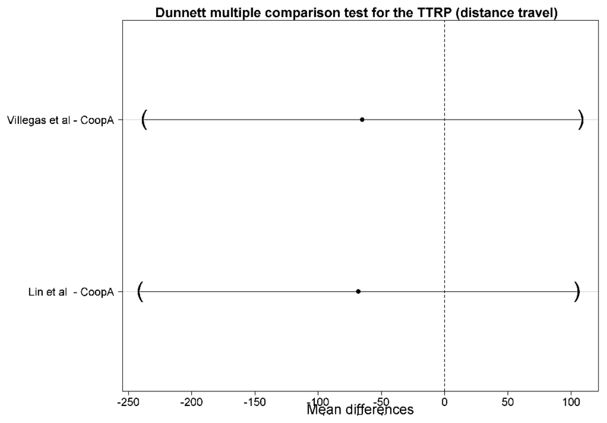

Fig. 4. Dunnett test

Table 1. Number of feasible routes and the best solution founded

Id instance	Number of routes	Best solution founded
1	1396	625.853
	4197	592.273
	7056	581.493
	9807	579.903
	12560	573.213
	14015	566.056

Villegas et al in [26] indicated that the set-partitioning model for the TTRP is often impractical. It is due to the huge number of feasible routes, and since it is impossible to compute all of them, the CoopA scheme builds a considerable number of them to tackle the aforementioned drawback. Table 1 details the number of routes computed by the CoopA scheme, and the best solution founded for the instance number one.

5 Conclusions and Future Research

The CoopA scheme detailed above, is suitable to tackle the TTRP. It is a well-known NP-hard issue. The main drawback of the set-partitioning model, i.e., the inability to compute all the routes, is cleverly resolved by devised procedures, and detailed in section III-B. Based on the results shown in section IV, the CoopA scheme is competitive. It was not necessary to incorporate auxiliary graphs to create feasible routes for those possible mix routes.

The set of instances used in the comparison are considered benchmarking. Therefore, the use of the Dunnett test is clearly justified and forceful. The performance of the CoopA scheme should be taken into account in the literature.

The proposal of the CoopA, i.e., considers all the members of the population to obtain a fitness for the all the population is substantial. Each member participates and cooperates to identify the fitness of the population. It is obtained by choosing from the routes of minimum total distance.

Although each member of the CoopA scheme only has partial information of the solution for the population, it is not a drawback for the CoopA scheme, on the contrary, this enriches its performance by consider many routes in the solution (see Table 1).

As future work, other greedy procedures should be implemented to create trips, to help the CoopA to find more suitable routes. In addition, other procedures should be incorporated to get offspring, to enhance the performance of the CoopA. Other optimization problems should be resolved by the CoopA scheme, in order to confirm its performance. Finally, application tools for users should be implemented in practice and real life situations using this approach.

Acknowledgments

A gratitude to all the reviewers for their comments in improving the manuscript.

References

1. **Absi, N., Cattaruzza, D., Feillet, D., Housseman, S. (2017).** A relax-and-repair heuristic for the swap-body vehicle routing problem. Annals of Operations Research, Vol. 253, pp. 957–978.
2. **Bodin, L., Levy, L. (2000).** Scheduling of local delivery carrier routes for the United States postal service. In Arc Routing: theory, solutions and applications. Springer, pp. 419–442.
3. **Boudia, M., Prins, C., Reghioui, M. (2007).** An effective memetic algorithm with population management for the split delivery vehicle routing problem. Hybrid Metaheuristics: 4th International Workshop, HM 2007, Dortmund, Germany, October 8-9, 2007. Proceedings 4, Springer, pp. 16–30.
4. **Chao, I.-M. (2002).** A tabu search method for the truck and trailer routing problem. Computers & Operations Research, Vol. 29, No. 1, pp. 33–51.
5. **Del Pia, A., Filippi, C. (2006).** A variable neighborhood descent algorithm for a real waste collection problem with mobile depots. International Transactions in Operational Research, Vol. 13, No. 2, pp. 125–141.
6. **Derigs, U., Pullmann, M., Vogel, U. (2013).** Truck and trailer routing—problems, heuristics and computational experience. Computers & Operations Research, Vol. 40, No. 2, pp. 536–546.
7. **Fathollahi-Fard, A. M., Ranjbar-Bourani, M., Cheikhrouhou, N., Hajiaghaei-Keshteli, M. (2019).** Novel modifications of social engineering optimizer to solve a truck scheduling problem in a cross-docking system. Computers & Industrial Engineering, Vol. 137, pp. 106103.
8. **Gerdessen, J. C. (1996).** Vehicle routing problem with trailers. European Journal of Operational Research, Vol. 93, No. 1, pp. 135–147.
9. **Hoff, A., Lokketangen, A. (2008).** A tabu search approach for milk collection in western norway using trucks and trailers. .
10. **Labadi, N., Prins, C., Reghioui, M. (2008).** A memetic algorithm for the vehicle routing problem with time windows. RAIRO-Operations research, Vol. 42, No. 3, pp. 415–431.
11. **Lin, S.-W., Vincent, F. Y., Chou, S.-Y. (2009).** Solving the truck and trailer routing problem based on a simulated annealing heuristic. Computers & Operations Research, Vol. 36, No. 5, pp. 1683–1692.
12. **Maghfiroh, M. F., Hanaoka, S. (2018).** Dynamic truck and trailer routing problem for last mile distribution in disaster response. Journal of Humanitarian Logistics and Supply Chain Management, Vol. 8, No. 2, pp. 252–278.
13. **Mart, R., Pardalos, P. M., Resende, M. G. (2018).** Handbook of heuristics. Springer Publishing Company, Incorporated.
14. **Prins, C. (2004).** A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & operations research, Vol. 31, No. 12, pp. 1985–2002.
15. **Prins, C., Lacomme, P., Prodhon, C. (2014).** Order-first split-second methods for vehicle

routing problems: A review. Transportation Research Part C: Emerging Technologies, Vol. 40, pp. 179–200.

16. **Renaud, J., Boctor, F. F., Laporte, G. (1996).** An improved petal heuristic for the vehicle routeing problem. Journal of the operational Research Society, Vol. 47, No. 2, pp. 329–336.
17. **Ryan, D. M., Hjorring, C., Glover, F. (1993).** Extensions of the petal method for vehicle routeing. Journal of the Operational Research Society, Vol. 44, No. 3, pp. 289–296.
18. **Scheuerer, S. (2006).** A tabu search heuristic for the truck and trailer routing problem. Computers & Operations Research, Vol. 33, No. 4, pp. 894–909.
19. **Semet, F. (1995).** A two-phase algorithm for the partial accessibility constrained vehicle routing problem. Annals of Operations Research, Vol. 61, pp. 45–65.
20. **Theophilus, O., Dulebenets, M. A., Pasha, J., Lau, Y.-y., Fathollahi-Fard, A. M., Mazaheri, A. (2021).** Truck scheduling optimization at a cold-chain cross-docking terminal with product perishability considerations. Computers & industrial engineering, Vol. 156, pp. 107240.
21. **Upton, G., Cook, I. (2014).** A dictionary of statistics 3e. Oxford quick reference.
22. **Vahrenkamp, R. (1989).** Transportation logistic in rural setting: the case of milk collection. Department of Business and Economics, University of Kassel, Vol. 5, pp. 1989.
23. **Villegas, J., Prins, C., Prodhon, C., Medaglia, A., Velasco, N. (2010).** GRASP/VND with path-relinking for the truck and trailer routing problem. TRISTAN VII (7th triennial symposium on transportation analysis).
24. **Villegas, J., Prins, C., Prodhon, C., Medaglia, A., Velasco, N. (2011).** Heuristic column generation for the truck and trailer routing problem. ROUTE 2011.
25. **Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L., Velasco, N. (2010).** Grasp/vnd and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots. Engineering Applications of Artificial Intelligence, Vol. 23, No. 5, pp. 780–794.
26. **Villegas, J. G., Prins, C., Prodhon, C., Medaglia, A. L., Velasco, N. (2013).** A matheuristic for the truck and trailer routing problem. European Journal of Operational Research, Vol. 230, No. 2, pp. 231–244.
27. **Wang, C., Zhou, S., Gao, Y., Liu, C. (2018).** A self-adaptive bat algorithm for the truck and trailer routing problem. Engineering Computations, Vol. 35, No. 1, pp. 108–135.
28. **Yu, V., Lin, T., Lu, C.-C. (2011).** An ant colony system for solving the truck and trailer routing problem. J. Chin. Inst. Transp, Vol. 23, pp. 199–218.
29. **Yuan, S., Fu, J., Cui, F., Zhang, X. (2020).** Truck and trailer routing problem solving by a backtracking search algorithm. Journal of Systems Science and Information, Vol. 8, No. 3, pp. 253–272.

Article received on 14/02/2024; accepted on 09/12/2024.
**Corresponding author is Luis Eduardo Urbán-Rivero.*