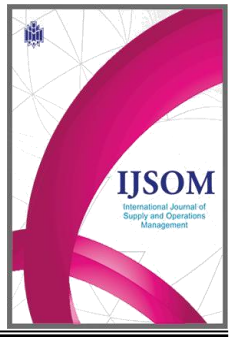
International Journal of Supply and Operations Management

IJSOM

November 2017, Volume 4, Issue 4, pp. 328-340
ISSN-Print: 2383-1359
ISSN-Online: 2383-2525
www.ijsom.com



An Enhanced Evolutionary Local Search for the Split Delivery Vehicle Routing Problem

Sanae Larioui [a,*]

[a] *ENSATE, University of Abdelmalek Essaadi, Mhannech II, Tetouan, Morocco*

Abstract
We present a simple and effective metaheuristic algorithm for the Split Delivery Vehicle Routing Problem (SDVRP). The SDVRP is a relaxation of the classical Vehicle Routing Problem in which a customer demand may be serviced by more than one vehicle. The objective is to find a set of least cost trips for a fleet of identical vehicles to service geographically scattered customers with or without splitting. The proposed method is a hybridization between a Variable Neighborhood Search (VNS), an Evolutionary Local Search (ELS) and a Variable Neighborhood Descent (VND). It combines the multi-start approach of VNS and ELS and the VND intensification and diversification strategies. This new method is tested on three sets of instances from literature containing a total of 77 benchmark problems. The obtained results show that the algorithm outperforms all previously published metaheuristics. 62 instances out of 77 are improved.

Keywords: Vehicle routing problem; Split delivery; Variable neighborhood search; Evolutionary local search; Variable neighborhood descent.

1. Introduction

The Vehicle Routing Problem (VRP) is a well-known combinatorial optimization problem with a high practical relevance. It consists in designing the optimal set of routes for a fleet of identical vehicles in order to serve a given set of customers, where each customer must be visited exactly once by a vehicle. The VRP has a wide range of applications such as in fuel distribution, bank deliveries, and postal deliveries and so on, and has been the subject of intensive research. The Split Delivery Vehicle Routing Problem (SDVRP) is a relaxation of the VRP in which a customer demand may be responded to by more than one vehicle. This problem has recently received a lot of attention from researchers. This is mainly because of the substantial savings in the total cost and the number of vehicles used that can be obtained by allowing split deliveries.

In some real-life applications, the customers' demand may exceed the vehicle capacity and then more than one vehicle is required to serve it. However, splitting demands can also be interesting when they are smaller than the vehicle capacity. The simplest example that illustrates such a case is shown in Figure 1. Even if all the customer demands (2 units for each customer) are smaller than the vehicle capacity (equal to 3), the solution where the demand of customer b is split has a better cost than the solution without splitting that uses one less vehicle.

This paper presents a simple and effective metaheuristic algorithm for the Split Delivery Vehicle Routing Problem (SDVRP). The objective is to find a set of trips with the least cost for a fleet of identical vehicles to service geographically scattered customers with or without splitting. The proposed method is a combination of a Variable Neighborhood Search

(VNS), an Evolutionary Local Search (ELS), and a Variable Neighborhood Descent (VND). It puts the multi-start approach of VNS and ELS and the VND intensification and diversification strategies together. This new method is tested on three sets of instances from the literature containing a total of 77 benchmark problems. The obtained results show that the algorithm outperforms all the previously published metaheuristics. 62 instances out of 77 are improved.

Corresponding author email address: sanae_lar@hotmail.com

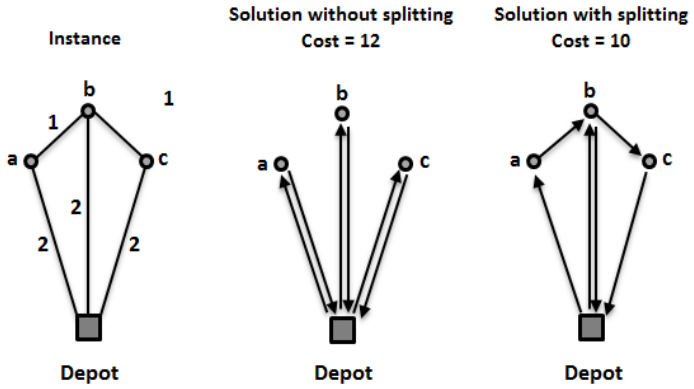


Figure 1. Example of the saving obtained by allowing the splitting

2. Review of the literature

The SDVRP was introduced by Dror and Trudeau (1989, 1990), who showed the advantage of splitting demands and proved the NP-hardness of the problem and some structural properties of optimal solutions. Later, Archetti et al. (2006) gave a bound on the cost reduction that can be obtained by allowing the splitting.

Three lower bounds are proposed for the SDVRP. The first one was obtained from a cutting plane algorithm by Belenguer et al. (2000). The second lower bound which is slightly better than the first one was proposed by Jin et al. (2008) who developed a column generation approach. The last one, recently presented by Moreno et al. (2010), is based on an algorithm that combines column and cut generation approaches and improves many of the best known lower bounds found by the two previous methods. Concerning exact solution algorithms for the SDVRP, Dror et al. (1994) proposed the first formulation of the problem, several families of valid inequalities, and a branch and bound procedure. A dynamic programming model with infinite state and action spaces was also presented by Lee et al. (2006) and its results were improved later by Jin et al. (2007) using a two-stage algorithm based on some new valid inequalities.

Heuristic approaches have been more widely investigated since the exact methods mentioned above cannot solve instances with more than 21 customers. A constructive heuristic for the SDVRP was presented by Dror and Trudeau (1990), and more sophisticated metaheuristics were proposed recently by several researchers. The first metaheuristic for the SDVRP is the tabu search algorithm of Archetti et al. (2006). The results of this method were improved by Boudia et al. (2007) using a memetic algorithm with population management (MA|PM), Mota et al. (2007) using a scatter search algorithm, and Chen et al. (2007) as well as Archetti et al. (2008) using solution approaches that combine heuristics and integer programming components. Recently, Aleman et al. (2010) proposed an adaptive memory algorithm coupled to a variable neighborhood descent which improves the results of the previous methods, except for the results of the MA|PM of Boudia et al. (2007).

Some extensions of the SDVRP were also studied. They mainly concern the consideration of time windows constraints or that of the arc routing version. The SDVRP with time windows (SDVRPTW) was considered in 1999 by Guéguen (xxxx) who proposed a formulation and adapted a column generation approach for the VRP with time windows to the SDVRPTW. This work was then improved in M. Gendreau (1999). The only metaheuristic dedicated to the SDVRPTW is the tabu search method of Ho and Haugland (2006). Frizzell and Giffin (1995) studied the SDVRPTW with grid network distances.

For the arc routing version (the Split Delivery Capacitated Arc Routing Problem, SDCARP), an adaptation of the SDVRP heuristic of Dror and Trudeau for the SDCARP with time windows was proposed by Mullaseril et al. (1997) to solve a livestock feed distribution problem. More recently, an important investigation of the SDCARP was presented in (2010, 2008). Two integer programming formulations, several constructive heuristics, two lower bounds and two metaheuristics were proposed. The authors also show that the savings that can be obtained by allowing the splitting in arc routing problems are similar to the ones obtained by Dror and Trudeau on the SDVRP although the length of a required edge is counted each time it is traversed for a partial service.

Our special interest lies in the combination of several components of three classical metaheuristics including the Variable Neighborhood Search (VNS), the Evolutionary Local Search (ELS), and the Variable Neighborhood Descent (VND). The resulting framework is applied to the SDVRP. The rest of this paper is organized as follows: Section 2 presents the problem and introduces some notations. The hybridization of the three metaheuristics mentioned above is described in Section 3. The components of the SDVRP are given in Section 4 and the experimental results are shown in Section 5. Finally, conclusions are drawn in the last section.

3. Statement of the problem

The SDVRP is defined on a complete weighted and undirected network $G = (N, E, C)$. N is a set of $n + 1$ nodes indexed from 0 onwards. Node 0 corresponds to a depot with identical vehicles of capacity W. Each other node i, $i = 1, 2, ..., n$ has a known demand q_i. The weight $c_{ij} = c_{ji}$ on each edge (i, j) of E is the travelling cost between nodes i and j. We assume that no demand q_i exceeds vehicle capacity W. Otherwise, for each customer i such that $q_i > W$, an amount of demand W can be deducted from q_i to build one dedicated trip with a full load until, as shown in [3], the residual demand fits the vehicle capacity.

We emphasize that partial deliveries are allowed, so some customers (called split customers) can be visited more than once. The objective is to determine a set of vehicle trips of minimum total cost. Each trip starts and ends at the depot and supplies a subset of customers. The number of trips or vehicles used is a decision variable. It is assumed that the triangle inequality holds: in that case, solutions in which each trip visits its customers only once dominate the others. In other words, if one customer is visited several times, these visits are done by distinct trips.
The problem can be formulated as follows:

Minimize

$$\sum_{k=1}^{M}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}.x_{ij}^{k} \qquad (1)$$

Subject to

$$\sum_{k=1}^{M}\sum_{i=1}^{n} x_{ij}^{k} \geq 1 \qquad \forall j \in V \qquad (2)$$

$$\sum_{i=1}^{n} x_{ip}^{k} - \sum_{j=1}^{n} x_{pj}^{k} = 0 \qquad \forall p \in V \;; \forall k \in F \qquad (3)$$

$$\sum_{k=1}^{M} y_{i}^{k} = 1 \qquad \forall i \in V \setminus \{1\} \qquad (4)$$

$$\sum_{i=2}^{n} q_i \, y_{i}^{k} \leq W \qquad \forall k \in F \qquad (5)$$

$$y_{i}^{k} \leq \sum_{j=1}^{Mn} x_{ji}^{k} \qquad \forall i \in V \setminus \{1\} \; ; \qquad \forall k \in F \qquad (6)$$

$$\sum_{i,j \in S} x_{ij}^{k} \leq |S[-1 \qquad S \subseteq V \setminus \{1\}; 2 \leq |S| \leq n-1; \forall k \in F \qquad (7)$$

$$x_{ij}^{k} \in \{0,1\} \qquad \forall \mathrm{i,j} \in V; \forall k \in F \qquad (8)$$

$$y_{i}^{k} \geq 0 \qquad \forall \mathrm{i,j} \in V; \forall k \in F \qquad (9)$$

Constraint (2) guarantees the service of each customer by at least one vehicle. Constraint (3) is the flood conservation constraint. The respect for customer demand and the capacity of the vehicles are ensured respectively by constraints (4) and (5). A point can be processed only by a vehicle passing through it and this is guaranteed by constraint (6). Finally, constraint (7) is the subtours elimination constraint.

4. General presentation of the solution approach

The metaheuristic proposed in this study is an enhanced version of a new method called hybrid GRASP × ELS that was applied successfully to the classical Vehicle Routing Problem by Prins in 2009.

The GRASP is a simple metaheuristic proposed by Feo and Bard (1989) in which each iteration consists of two phases: constructing a greedy randomized solution and improving it with a local search procedure. The solution produced in the first phase is generated by adding elements to the problem's solution chosen randomly from a list of well ranked candidate elements according to a greedy function. The GRASP has been applied to a wide range of combinatorial optimization problems, and is often faster but less effective than other metaheuristics like tabu search or genetic algorithms.

Algorithm 1-ELS algorithm :

```
Sol=Heuristic();
for gen :=1 to ngen do
    BestChildSol :=Sol;
    for son :=1 to nchild do
        ChildSol := Sol;
        Perturb(ChildSol);
        ChildSol := LocalSearch(ChildSol);
        if Cost(ChildSol) < Cost(BestChildSol) then
            BestChildSol := ChildSol;
        end if
    end for
    if Cost(BestChildSol)< Cost(Sol) then
        Sol=BestChildSol;
    end if
end for
```

Evolutionary Local Search (ELS) was introduced by Wolf and Merz (2007) to solve a telecommunication problem. Starting with a good solution built by a heuristic and improved with a local search, ELS performs *ngen* generations. Each generation builds *nchild* children-solutions. Each child is obtained taking a copy of the incumbent solution *Sol*, which is then modified by a perturbation procedure and improved by local search. If the best offspring solution improves the incumbent solution *Sol*, it replaces it for the next generation. The method is summarized in Algorithm 1. An Iterated Local Search is obtained if only one offspring lsolution is built in each generation (*nchild* = 1).

In the hybrid GRASP × ELS, each of the *niter* GRASP iterations generates one starting solution and applies ELS to it. The advantage of this hybridization is the possibility of getting out of local optima when the ELS converges and restarts the local search from a new solution. Reghioui (2008) proposed a similar method that uses several greedy randomized heuristics in the first phase. Its application to the Capacitated Arc Routing Problem with Split Deliveries has resulted in high-quality solutions compared with those obtained from a memetic algorithm. The general structure of the hybrid GRASP × ELS is given in Algorithm 2.

Algorithm 2 - Hybrid GRASP x ELS algorithm:

```
for i:=1 to niter do
    Sol := Greedy_Randomized_Heuristic();
    Sol := ELS(Sol);
    If Cost(Sol) < Cost(BestSol) then
        BestSol := Sol;
    end if
end for
```

A possible target of the improvement of the hybrid GRASP × ELS would be the quality of solutions produced in the first phase. Each iteration of the method can be seen as a separate ELS that does not benefit from the information collected about the best solution in the previous iterations. In addition to that, when the solutions generated by the greedy randomized heuristic are of poor quality, the local search takes a long time to converge to local optima. The enhancement proposed by the method sketched in Algorithm 3 consists of replacing the greedy randomized heuristic in the first phase by a procedure that modifies the best solution achieved so far, inspired by the strategy used in the shaking phase of the Variable Neighborhood Search (VNS).

The main principle of VNS is the systematic change of neighborhood within the search. Let (N_1,...,N_{kmax}) denote a finite set of neighborhoods, where $N_k(S)$ is the set of solutions in the k^{th} neighborhood of *S*. VNS escapes from a local optima by initiating the search process from the starting points sampled from one of the k_{max} neighborhoods of the best solution.

A similar strategy is used in the approach proposed for the SDVRP. When the ELS converge, a new solution is drawn randomly in a neighborhood N_k to change the search region. This new solution often keeps some characteristics of the best solution, thus the intensification phase takes less time to achieve a local optima. Another key point in the efficiency of the method is the use of Variable Neighborhood Descent (VND) as the local search in the ELS phase. More details about the VND are given in the next section. The solution approach proposed for the SDVRP is called the enhanced evolutionary local search (EELS) in the sequel.

Algorithm 3 - Enhanced evolutionary local search algorithm:

```
Sol := Heuristic();
k:=1;
BestSol := Sol;
for i:=1 to ni do
    Sol := Neighborhood_k(BestSol);
    Sol := ELS(Sol);
    If Cost(Sol) < Cost(BestSol) then
        BestSol := Sol;
        k := 1;
    else
        k := min(k_max, k + 1);
    end if
end for
```

5. Components of the SDVRP

5.1 Initial Solutions

The EELS proposed for the SDVRP uses two classical VRP heuristics and one SDVRP heuristic to generate the first three initial solutions: the savings heuristic of Clarke and Wright (1964), the Sweep heuristic of Gillet and Miller (1974), and the Split-Insertion Heuristic (SIH) of Reghioui (2008). The remaining initial solutions are drawn randomly from three neighborhoods around the best solution.

The saving algorithm starts with assigning a trip to each customer. Then, at each iteration it concatenates two distinct trips whose total saving is the largest until additional concatenations would violate vehicle capacity or increase total cost. In the sweep heuristic, clusters of customers compatible with the vehicle capacity are generated by the rotation of a half-line centered on the depot. A vehicle route is then computed in each cluster by solving a traveling salesman problem.

SIH is an efficient heuristic able to split demands. It starts with a solution reduced to a dummy loop on the depot. At each iteration, the heuristic evaluates all non-routed customers and inserts with or without splitting the one that has the minimum insertion cost. For each customer, the best subset of trips is chosen for the insertion. New trips are created when no further insertions are possible in the current trips.

The initial solution of each iteration $i > 4$ is obtained as follows: first, the best solution is temporarily converted into a giant tour by concatenating the sequences of customers of its trips. Random moves are then executed and the giant tour is finally converted back into an SDVRP solution, using a splitting procedure (SPLIT) proposed by Boudia et al. (2007) in their MA|PM.

Three moves are used to modify the giant tour of the best solution (*BGT*, Best Giant Tour):

1. M_1: Swap two customers.
2. M_2: Relocate one customer.
3. M_3: Invert a chain of customers (2-opt).

In the fourth iteration, a new giant tour *GT* is obtained by applying M_1 to *BGT*. *SPLIT* is then used to convert *GT* into a feasible solution *S*. If the improvement of S with the ELS phase does not give a better solution than the best one, M_2 is used in the next iteration. Otherwise, M1 is considered.

In general, at each iteration $i > 4$, M_1, M_2 and M_3 are used respectively after one, two and three or more iterations without improving the best solution found so far.

5.2 Local search

The local search used in the ELS phase of the EELS is structured as a Variable Neighborhood Descent (VND). VND [22] is a local search technique concerned with systematic neighborhood changes. It is based on the observation that a local minimum for a given neighborhood is not necessarily a local minimum for another one. Let the set of neighborhoods be (N_1, ...,N_{kmax}). Starting from $k = 1$, at each iteration the VND performs a local descent based on neighborhood N_k. If the obtained local optimum is better than the best current solution, then it replaces it and k is set to 1. Otherwise k is incremented. The method stops when no further improvement can be found with the neighborhood N_{kmax}.

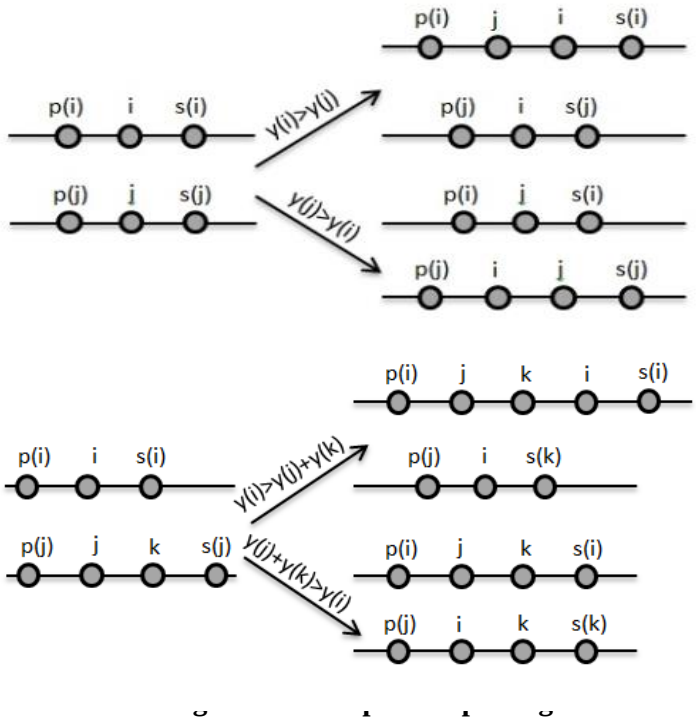
Two neighborhoods are used in our VND. The first one contains moves involving one or two trips and is defined by three classical moves of the VRP (Or-Opt which relocates a chain of 1 to 3 customers, Exchange which swaps two chains

of 1 to 3 customers and 2-Opt) and two other moves that exchange parts of customers' demand, introduced by Boudia et al. (2007). Figure 2 taken from the same study shows an example of these two moves which are able to split demands of customers. They are defined as follows: the first move exchanges two customers *i* and *j* pertaining to two distinct trips and splits the one with the largest demand. The second move is similar but exchanges a customer *i* with two successive customers *j* and *k* pertaining to another trip.

The second neighborhood contains a move operating on more than two trips. At each iteration it removes one customer from all the routes in which it is serviced and performs a best insertion with or without splitting. This last move is an improvement of the K-split interchange operator of Dror and Trudeau (1989), which was proposed in (2007). Figure 1 shows an example of this move. Customer b is removed from its route (solution without splitting) and inserted with splitting in the other two routes (solution with splitting).

5.3 Perturbation Procedure

The perturbation procedure is called during the ELS iterations. The move M_3, which is based on 2-opt, is used to generate children-solutions. The procedure follows the same process as the one used in the first phase to generate initial solutions by perturbing giant tours. The solution *S* passed to ELS is first converted into a giant tour, some 2-opt moves are then executed and the resulting tour is converted back into an SDVRP solution. The objective here is to generate small perturbations in order to explore the local region around the solution *S*.



5.4 General structure

The EELS is summarized in algorithm 4. The first phase (from line 3 to line 19) generates the initial solution of each iteration *i*. Constructive heuristics CW, GM and SIH are used in the first three iterations. In the remaining iterations, the trips of the best solution *BestSol* are concatenated to give a giant tour. Then, a random solution in the neighborhood of *BestSol* is obtained as follows: procedures *PerturbSwap* and *PerturbRelocate* are called respectively when the number of iterations without improvement *IterWI* is equal to 1 and 2 and Perturb2Opt is called when it is greater than or equal to 3. Each one of them executes *pmax* moves and the resulting tour is converted into an SDVRP solution Sol and improved by the VND.

The second, or ELS, phase (from line 20 to line 46) generates several child solutions by perturbing a copy of *Sol* with procedure *Perturb2Opt* and improving it with the VND. The number of moves *p* executed by *Perturb2Opt* is initialized to a minimum value *pmin* at the beginning of the ELS or when a generation improves the incumbent solution *Sol*. If *Sol* is not improved *p* is incremented in order to escape from the local optima, but without exceeding a maximum value *pmax*. At the end of each generation, the best child solution *BestChildSol* replaces *Sol* in case of improvement. The best solution *BestSol* is updated after the end of the ELS generations.

The originality of this implementation compared to the classical one is that it limits the number of iterations without improvement instead of the number of iterations. The main advantage of this strategy is that the search process is not

stopped when improvements are obtained from the last iterations (which means that the search region is promising). The same strategy is also used for generations and children. The EELS stops when the number of iterations without improvement (*IterWI*) of the best solution *BestSol* exceeds a maximum value *MaxNIWI*. Similarly, the ELS phase stops after *MaxNGWI* generations without improvement of the incumbent solution *Sol*, and no further child solution is generated when the best child solution *BestChildSol* cannot be improved after *MaxNC WI* perturbations of *Sol*.

6. Computational results

6.1 Implementation and instances

In this section we report the performance of our method on the SDVRP described in Section 3. All the algorithms were coded in Delphi and executed on a Pentium IV 3 GHz with 1 GB of RAM under Windows XP. The tests were performed on well-known instances and compared to both the best lower bounds when available and the best known heuristic solutions in the related literature. Three sets of instances with a total of 77 SDVRP benchmark problems were used.

The first set contained 42 instances introduced by Archetti et al. (2006) to evaluate the performances of a tabu search approach (2006) and an optimization based algorithm (2008). These instances were derived from 7 CVRP test problems described by Christofides et al. by randomly generating the demands of customers using five different intervals with respect to vehicle capacity W. In addition to the original instances, five new groups of 7 instances were obtained using the following intervals: [0.1*W*,0.3*W*], [0.1*W*,0.5*W*],[0.1*W*,0.9*W*],[0.3*W*,0.7*W*],[0.7*W*,0.9*W*]. The number of customers in these instances ranges from 50 to 199.

The second set included 14 randomly generated instances from (2000) with known lower bounds (2000, 2007, 2010). They contained 50 to 100 customers. The demand of customers was generated using the same strategy as Archetti et al. [3] and the intervals [0.1*W*,0.9*W*], [0.3*W*,0.7*W*] and [0.7*W*,0.9*W*].

The last comparison was conducted on a set of 21 instances proposed by Chen et al. [8] that have between 8 and 288 customers. The demands were obtained using the interval [0.7*W*,0.9*W*]. The specificity of these instances lies in their structure. In each problem, customers are located in concentric circles around the depot in a manner that allows a visual estimation of a near-optimal solution.

The appropriate adjustment of parameters in our algorithm can make a significant difference in terms of speed and solutions quality. The EELS have 5 parameters that were tuned in order to obtain satisfactory results from the three sets of instances used. The following setting was considered: *MaxNIWI* = 10 , *MaxNGWI* = 30, *MaxNCWI* = 5, *pmin* = 1 and *pmax* = 4.

6.2 Results

Table 1 shows the results of the EELS on the first set of instances and the best published results obtained so far by SplitTabu and the optimization based algorithm (Opt−Based) proposed in (2006, 2008). The Opt−Based columns give the percentage of improvement over the solution of SplitTabu and the computational time. The EELS columns provide the exact costs obtained from using one run and the setting given above.

The results indicate that the EELS improves 41 out of the 42 tested instances, while being 2 times faster than the optimization based algorithm. Moreover, the average percentage of improvement over the solution obtained with SplitTabu is 2.51% for the EELS versus 0.55% for the optimization based algorithm.

For the instances of Belenguer et al. (2000), exact methods and some heuristic approaches consider integer distances by rounding them to the nearest integer while other heuristics use real numbers. To be able to compare all these methods, both real and integer distances have been used.

Algorithm 4 :General structure of the mix metaheuristic

```
I :=1; IterW I := 0
while IterW I <= MaxN IW I do
    if  i = 1, 2, 3 then
        Sol := CW(), GM(), SIH(); i := i + 1
    else
        Concat (BestSol, Tour)
        if IterW I = 1, 2 then
            PerturbSwap(Tour,pmax), PerturbRelocate(Tour,pmax)
    end if
    if  IterW I > 2 then
        Sol := Perturb2Opt(Tour,pmax)
    end if
    Split(Tour,Sol)
end if
Sol := VND(Sol)
Concat (Sol,Tour)
p := pmin; GenW I := 0
while GenW I < MaxNGW I do
    BestChildSol  := Sol; ChildW I := 0
    while ChildW I <= MaxNCW I do
        ChildTour := Tour
        Perturb2Opt(ChildTour,p)
        Split(ChildTour,ChildSol)
        ChildSol := VND(ChildSol)
        if Cost(ChildSol) < Cost(BestChildSol) then
            BestChildSol := ChildSol; ChildW I := 0
        else
            ChildW I:=ChildW I+1
        end if
    end while
    if Cost(BestChildSol) < Cost(Sol) then
        Sol := BestChildSol
        Concat(sol,Tour)
        p := pmin,GenW I := 0
    else
        p := min(pmax,p+1);GenW I := GenW I+1
    end if
end while
if Cost(Sol) < Cost(BestSol) then
    BestSol := Sol; IterW I := 0
else
    IterW I := IterW I+1
```

Table 1. Computational results of the SDVRP instances of archetti

			Opt-Based		EELS		
File	Demand	Split Tabu	Saving	Time	Cost	saving	Time
p1-50		3 307 907	0,59	97	5 246 111	1,16	29,50
p2-75		8 542 757	0,08	52	8 238 883	3,56	125,84
p3-100		8 413 577	0,01	51	8 273 926	1,66	399,38
p4-150		10 708 613	0,70	298	10 284 995	3,96	642,59
p5-199		13 403 505	0,15	297	13 105 908	2,22	1 362,38
p6-120		13 403 505	0,15	298	13 105 908	2,22	1 361,89
p7-100		10 569 587	0,00	262	10 385 320	1,74	510,16
p1-50	0.1-0.3	7 653 121	0,37	256	7 721 860	-0,90	53,84
p2-75	0.1-0.3	11 340 760	0,99	161	11 141 998	1,75	148,95
p3-100	0.1-0.3	15 151 732	0,65	159	14 701 795	2,97	254,97
p4-150	0.1-0.3	21 018 042	0,24	1152	20 426 855	2,81	716,69
p5-199	0.1-0.3	25 858 494	0,13	567	24 970 978	3,43	1 665,02
p6-120	0.1-0.3	25 858 494	0,13	545	24 970 978	3,43	1 665,55
p7-100	0.1-0.3	30 604 668	1,41	585	29 116 951	4,86	577,70
p1-50	0.1-0.5	10 391 059	0,30	866	10 058 547	3,20	49,45
p2-75	0.1-0.5	15 566 936	0,53	646	15 051 288	3,31	109,75
p3-100	0.1-0.5	20 541 296	1,46	201	20 245 591	1,44	332,94
p4-150	0.1-0.5	29 916 416	0,49	517	28 873 354	3,49	635,05
p5-199	0.1-0.5	36 242 004	0,84	1138	35 081 493	3,20	1 783,37
p6-120	0.1-0.5	36 242 004	0,84	1114	35 081 493	3,20	1 783,62
p7-100	0.1-0.5	45 026 152	0,59	365	42 258 299	6,15	833,47
p1-50	0.1-0.9	15 119 826	0,08	2939	14 918 496	1,33	47,41
p2-75	0.1-0.9	23 386 654	0,04	361	23 156 274	0,99	120,80
p3-100	0.1-0.9	31 552 228	0,60	620	31 071 622	1,52	395,97
p4-150	0.1-0.9	46 741 320	0,31	592	45 935 431	1,72	1 208,69
p5-199	0.1-0.9	57 158 484	0,10	806	55 899 746	2,2	2 015,67
p6-120	0.1-0.9	57 158 484	0,10	813	55 899 746	2,2	2 015,00
p7-100	0.1-0.9	73 501 136	3,27	4882	69 022 946	6,09	463,50
p1-50	0.3-0.7	15 039 466	0,13	1684	14 918 503	0,8	43,41
p2-75	0.3-0.7	22 935 488	1,08	2551	22 474 891	2,01	118,72
p3-100	0.3-0.7	30 709 048	0,50	1605	30 183 767	1,71	345,12
p4-150	0.3-0.7	44 968 584	0,70	251	43 879 382	2,42	1 001,05
p5-199	0.3-0.7	55 711 292	0,38	1702	54 599 173	2,00	2 289,58
p6-120	0.3-0.7	55 711 292	0,38	1704	54 599 173	2,00	2 202,28
p7-100	0.3-0.7	71 682 608	0,58	7147	66 889 077	6,69	463,39
p1-50	0.7-0.9	21 736 308	0,06	834	21 690 614	0,21	98,69
p2-75	0.7-0.9	32 853 678	0,34	1872	32 261 032	1,80	256,08
p3-100	0.7-0.9	44 707 136	0,41	2433	43 988 851	1,61	399,28
p4-150	0.7-0.9	64 821 904	0,30	2460	64 166 262	1,01	1 138,45
p5-199	0.7-0.9	83 921 136	0,44	656	82 160 719	2,10	1 541,55
p6-120	0.7-0.9	83 921 136	0,44	656	82 160 719	2,10	1638,34
p7-100	0.7-0.9	106 733 056	2,34	3948	102 531 713	3,94	809,17
Average			**0,55**	**1193,88**		**2,51**	**800,86**

Tables 2 and 3 compare the performances of two versions of the EELS with the MA|PM of Boudia et al. (2007) and the algorithm of Aleman et al. (2010) noted ICA+VND on the instances of Belenguer et al. (2000). Both methods considered integer numbers. The first version uses the setting of parameters given above while in the second (fast version) fewer generations are considered in the ELS phase (5 instead of 30). Table 4 presents a comparison of the EELS with the heuristic of Chen et al. (2007) who tested their method on only five of these instances and used real numbers. Aleman et al. (2010) also executed their ICA+VND without rounding the euclidean distances. Their results for these five instances are sketched in the same table. The column LB in the three tables (2, 3 and 4) shows the lower bound obtained by Moreno et al. [23] since it is better than those of Belenguer et al. (2000) and Jin et al. (2007) on the 14 instances.

Compared to the MA|PM of Boudia et al. (2007), the slow version of the EELS finds better solutions on 9 instances and gives the same solution costs for 4 instances. The MA|PM outperforms the EELS only on one instance. The average deviation of the lower bound is 2.33% for the MA|PM and 1.82% for the EELS, however on average the EELS requires two minutes to find a solution while the MA|PM needs only 18 seconds. The fast version was designed to tackle this weakness. The number of generations without improvement has been reduced in order to make the EELS more competitive in terms of the computational time. Table 3 shows clearly that this version of the EELS is as fast as the MA|PM and also gives a slightly better average deviation of the LB. The number of improved solutions here is 6 instead of 9. The first version of EELS is slower than ICA+VND, but the second version is two times faster. Both versions outperform ICA+VND on all instances with regard to the solution quality.

Table 2. Results of the random SDVRP instances of Blenguer

		ICA+VND		MA| PM		EELS		
File	LB	Cost	Time	Cost	Time	Cost	Time	Best
S51D1	454,0	469	4,53	458	8,77	458	28,12	458
S51D2	694,2	726	4,05	707	7,44	705	24,94	704
S51D3	930,7	994	2,5	945	7,84	945	24,16	945
S51D4	1 539,0	1 700	2,89	1 578	11,98	1 570	33,36	1 570
S51D5	1 313,4	1 399	1,8	1 351	16,72	1 335	37,62	1 335
S51D6	2 141,7	2 221	2,27	2 182	9,92	2 170	73,37	2 170
S76D1	586,6	603	63,55	592	15,23	592	106,41	592
S76D2	1 061,1	1165	7,73	1 089	30,50	1 089	79,48	1 087
S76D3	1 395,9	1 485	12,23	1 427	12,89	1 422	83,25	1 420
S76D4	2 046,1	2 205	6,91	2 117	8,76	2 096	108,17	2 081
S101D1	704,9	757	210,36	717	49,75	716	318,03	716
S101D2	1 337,1	1 431	26,2	1 372	31,72	1 376	150,55	1 372
S101D3	1 832,2	1 975	27,84	1 891	33,98	1 868	250,84	1 868
S101D5	2 737,1	2 985	18,36	2 854	18,66	2 798	264,58	2 798
Average		**6,67**	**27,94**	**2,33**	**18,87**	**1,71**	**113,06**	**2,10**

The comparison with the heuristic of Chen et al. (2007) given in Table 4 confirms the superiority of the EELS which always finds better solutions while remaining 3 times faster. ICA+VND has the best computational time but gives solutions of poor quality.

The last comparison was performed on 21 instances proposed by Chen et al. (2007). The authors also compared their method to SplitTabu but their results are not included in Table 1 since they use different instances (generated following the same strategy as Archetti et al. (2006), but the demands of customers are different since the random generators are not initialized in the same way). This time the EELS finds 12 new best solutions. It outperforms the method of Chen et al. [8] on 18 instances and is twice as fast. ICA+VND is always the fastest method, however the EELS finds better results for 12 instances and gives a better average deviation of the lower bound, i.e. 1.03% instead of 1.48%. The EELS retrieves the optimum for 6 instances and is outperformed only on three (SD1, SD8 and SD9).

Table 3. Results of the random SDVRP instances of Blenguer (Fast version)

		ICA+VND		MA| PM		EELS		
File	LB	Cost	Time	Cost	Time	Cost	Time	Best
S51D1	454,0	469	4,53	458	8,77	458	4,72	458
S51D2	694,2	726	4,05	707	7,44	705	4,94	704
S51D3	930,7	994	2,5	945	7,84	945	7,06	945
S51D4	1 539,0	1 700	2,89	1 578	11,98	1 575	9,75	1 570
S51D5	1 313,4	1 399	1,8	1 351	16,72	1 340	6,22	1 335
S51D6	2 141,7	2 221	2,27	2 182	9,92	2 187	7,28	2 170
S76D1	586,6	603	63,55	592	15,23	592	19,14	592
S76D2	1 061,1	1165	7,73	1 089	30,50	1 097	17,83	1 087
S76D3	1 395,9	1 485	12,23	1 427	12,89	1 437	15,83	1 420
S76D4	2 046,1	2 205	6,91	2 117	8,76	2 081	25,52	2 081
S101D1	704,9	757	210,36	717	49,75	716	31,06	716
S101D2	1 337,1	1 431	26,2	1 372	31,72	1 376	36,06	1 372
S101D3	1 832,2	1 975	27,84	1 891	33,98	1 905	18,92	1 868
S101D5	2 737,1	2 985	18,36	2 854	18,66	2 835	21,77	2 798
Average		**6,67**	**27,94**	**2,33**	**18,87**	**2,25**	**16,15**	**2,10**

Table 4. Comparison with Chen et al.'s algorithm on the instances of Blenguer

		ICA+VND		EMIP+VRTR		EELS	
File	LB	Cost	Time	Cost	Time	Cost	Time
S51D4	1 549,7	1 708,00	2,89	1 586,5	201,74	1 564,52	73,73
S51D5	1 318,9	1 404,54	1,80	1 355,5	201,62	1 334,55	61,12
S51D6	2 154,7	2 230,06	2,27	2 197,8	301,90	2 180,42	77,28
S76D4	2 059,8	22 020,87	6,91	2 136,4	601,92	2 087,32	196,08
S101D5	2 767,6	2 999,31	18,36	2 846,2	645,99	2 806,02	358,27
Average		**6,67**	**27,94**	**2,74**	**390,63**	**1,21**	**153,3**

7. Conclusion

Based on the findings of this study we can say briefly that the combination of several components of different metaheuristics can lead to a strong and robust approach with a good alternation between diversification and intensification phases. The enhanced evolutionary local search proposed for the Split Delivery Vehicle routing Problem (SDVRP) combines diversification strategies of the Variable Neighborhood Search and the Evolutionary Local Search by using shaking techniques and perturbation procedures with the intensification method of the Variable Neighborhood Descent. The resulting approach has been compared to a lower bound and to the best four published metaheuristics for the SDVRP. Computational results confirm the efficiency of this new method as more than 80% of the tested instances were improved.

Table 5. Results of the SDVRP instances of Chen

		Chen et al.		ICA+VND		EELS	
File	LB	Cost	Time	Cost	Time	Cost	Time
SD1	228,3	228,28	0.7	228,28*	0,06	240,00	2,06
SD2	708,3	714,40	54,40	708,28*	0,22	708,28*	4,61
SD3	430,6	430,61	67,30	430,58*	0,17	430,58*	3,67
SD4	631,0	631,06	400,00	635,84	0,55	631,05*	5,70
SD5	1 390,6	1 408,12	402,70	1390,57*	0,69	1390,57*	10,81
SD6	831,2	831,21	408,30	831,24*	0,94	831,24*	8,56
SD7	3 640,0	3 714,40	403,20	3640,00*	1,03	3640,00*	15,67
SD8	5 068,3	5 200,00	404,10	5068,28*	1,75	5 080,00	27,50
SD9	2 044,2	2 059,84	404,30	2 071,03	2,91	2 062,09	53,81
SD10	2 684,9	2 749,11	400,00	2 747,83	3,58	2 704,05	89,13
SD11	13 275,0	13 612,12	400,10	13 280,00	3,97	13 280,00	80,56
SD12	7 175,0	7 399,06	408,30	7 279,97	4,00	7 223,63	88,84
SD13	10 053,6	10 367,06	404,50	10 110,58	5,80	10 110,58	218,81
SD14	10 588,2	11 023,00	5 021,70	10 893,50	15,49	10 741,83	244,75
SD15	14 908,5	15 271,77	5 042,30	15 168,28	18,33	15 128,48	401,94
SD16	3 381,3	3 449,05	5 014,70	3 635,27	39,71	3 440,62	769,87
SD17	26 317,2	26 665,76	5 023,60	26 559,93	17,42	26 503,59	705,55
SD18	14 029,2	14 546,58	5 028,60	14 440,59	40,38	14 220,29	916,73
SD19	19 707,2	20 559,21	5 034,20	20 191,19	27,64	20 032,24	793,25
SD20	39 252,8	40 408,22	5 053,00	39 813,49	63,18	39 666,02	1 516,98
SD21	11 271,0	11 491,67	5 051,00	11 799,60	738,49	11 467,08	4 290,78
Average		**1,98**	**2115,57**	**1,48**	**46,97**		**488,08**

References

B.E. Gillett and L.R. Miller (1974). A heuristic algorithm for the vehicle dispatch problem. *Operations Research*, Vol. 22, pp. 340–349.

C. Archetti, M.W.P. Savelsbergh, and M.G. Speranza (2006). Worst-case analysis for split delivery vehicle routing problems. *Transportation Science*, Vol. 40(2), pp. 226–234.

C. Archetti, M.G. Speranza, and A. Hertz (2006). A tabu search algorithm for the split delivery vehicle routing problem. *Transportation Science*, Vol. 40(1), pp. 64–73.

C. Archetti, M.G. Speranza, and M.W.P. Savelsbergh. (2008) An optimization based heuristic for the split delivery vehicle routing problem. *Transportation Science*, Vol. 42(1), pp. 22–31.

C. Gu´eguen (1999). Exact solution methods for vehicle routing problems. PhD thesis, Central School of Paris, France, (in French).

C. Prins (2009). Bio-inspired Algorithms for the Vehicle Routing Problem, chapter A GRASP × evolutionary local search hybrid for the vehicle routing problem, pp. 35–53. Springer.

E. Mota, V. Campos, and A. Corber´an (2007). A new metaheuristic for the vehicle routing problem with split demands. In C. Cotta and J. Van Hemert, editors, Evolutionary computation in combinatorial optimization, Lecture Notes in *Computer Science*, Vol. 4446, pp. 121–129, Berlin, Springer.

G. Clarke and J.W Wright. (1964), Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, Vol. 12, pp. 568–581.

J.M. Belenguer, E. Benavent, N. Labadi, C. Prins, and M. Reghioui (2010). Split delivery capacitated arc-routing problem: Lower bound and metaheuristic. *Transportation Science*, Vol. 44(2), pp. 206–220.

J.M. Belenguer, M.C. Martinez, and E. Mota (2000). A lower bound for the split delivery vehicle routing problem. *Operations Research*, Vol. 48(5), pp. 801–810.

L. Moreno, M. Poggi de Arag˜ao, and E. Uchoa (2010). Improved lower bounds for the split delivery vehicle routing problem. Operations Research Letters, Vol. 38, pp. 302–306.

M. Boudia, C. Prins, andM. Reghioui (2007). An effective memetic algorithm with population management for the split-delivery vehicle routing problem. In T. Bartz- Beielstein et al., editor, Hybrid Metaheuristics, Lecture Notes in Computer Science, Vol. 4771, pp. 16–30, Berlin. Springer.

M. Dror and P. Trudeau (1989). Savings by split delivery routing. *Transportation Science*, Vol. 23(2), pp. 141–145.

M. Dror and P. Trudeau (1990). Split delivery routing. Naval Research Logistics, Vol. 37, pp. 383–402.

M. Dror, G. Laporte, and P. Trudeau (1994). Vehicle routing with split deliveries. *Discrete Applied Mathematics*, Vol. 50, pp. 239–254.

M. Gendreau, P. Dejax, D. Feillet, and C. Gueguen (2006). Vehicle routing with time windows and split deliveries. Technical Report 2006–851, Laboratoire d'Informatique d'Avignon.

M. Jin, K. Liu, and R. Bowden (2007). A two stage algorithm with valid inequalities for the split delivery vehicle routing problem. *International Journal of Production Economics*, Vol. 105, pp. 228–242.

M. Jin, K. Liu, and B. Eksioglu (2008). A column generation approach for the split delivery vehicle routing problem. *Operations Research Letters*, Vol. 36, pp. 265–270.

M. Reghioui. Vehicle routing problems with time windows or split deliveries. PhD thesis, University of Technology of Troyes, 2008 (in French).

N. Mladenovi´c and P. Hansen (1997). Variable neighborhood search. *Computers & Operations Research*, Vol. 24(11), pp. 1097–1100.

P.A. Mullaseril, M. Dror, and J. Leung (1997). Split-delivery routing heuristics in livestock feed distribution. *Journal of the Operational Research Society*, Vol. 48(2), pp. 107–116.

P.W. Frizzell and J.W. Giffin(1995). The split delivery vehicle scheduling problem with time windows and grid network distances. Computers and Operations Research, Vol. 22(6), pp. 655–667.

S. Chen, B.L. Golden, and E. Wasil (2007). The split delivery vehicle routing problem: applications, algorithms, test problems and computational results. *Networks*, Vol. 49 (4), pp. 318–329.

S.C. Ho and D. Haugland (2004). A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. *Computers and Operations Research*, Vol. 31, pp.1947–1964.

S. Wolf and P. Merz (2007). Evolutionary local search for the super-peer selection problem and the p-hub median problem. In T. Bartz-Beielstein et al., editor, Hybrid Metaheuristics, *Lecture Notes in Computer Science*, Vol. 4771, pp. 1–15, Berlin,. Springer.

T.A. Feo and J. Bard (1989). Flight scheduling and maintenance base planning. *Management Science*, Vol. 35(12), pp. 1415–1432.

R.E. Aleman, X. Zhang, and R.R. Hill (2010). An adaptive memory algorithm for the split delivery vehicle routing problem. *Journal of Heuristics*, Vol. 16 (3), pp. 441–473.