Profitable Vehicle Routing Problem with Multiple Trips: Modeling and Variable Neighborhood Descent Algorithm

Ahlem Chbichib1,*, Racem Mellouli2, Habib Chabchoub3

1Department of Quantitative Methods, Faculty of Economic Sciences and Management, Sfax, 3018, Tunisia
2Department of computer and quantitative methods, the Higher School of Management, Sfax, 3018, Tunisia
3Department of Quantitative Methods, Institute of the High Commercial Studies, Sfax, 3018, Tunisia

Abstract In this paper, we tackle a new variant of the Vehicle Routing Problem (VRP) which combines two known variants namely the Profitable VRP and the VRP with Multiple Trips. The resulting problem may be called the Profitable Vehicle Routing Problem with Multiple Trips. The main purpose is to cover and solve a more complex realistic situation of the distribution transportation. The profitability concept arises when only a subset of customers can be served due to the lack of means or for insufficiency of the offer. In this case, each customer is associated to an economical profit which will be integrated to the objective function. The latter contains at hand the total collected profit minus the transportation costs. Each vehicle is allowed to perform several routes under a strict workday duration limit. This problem has a very practical interest especially for daily distribution schedules with limited vehicle fleets and short course transportation networks. We point out a new discursive approach for profits quantification which is more significant than those existing in the literature. We propose four equivalent mathematical formulations for the problem which are tested and compared using CPLEX solver on small-size instances. Optimal solutions are identified. For large-size instance, two constructive heuristics are proposed and enhanced using Hill Climbing and Variable Neighborhood Descent algorithm based on a specific three-arrays-based coding structure. Finally, extensive computational experiments are performed including randomly generated instances and an extended and adapted benchmark from literature showing very satisfactory results.

Keywords Profitable Vehicle Routing Problem, Multiple Trips, Mixed Integer Linear Programming, Constructive Heuristics, Hill Climbing, Variable Neighborhood Descent

1. Introduction

Nowadays, effective management of the supply chain is recognized as a determinant key of competitiveness and success for manufacturing and distribution organizations. The problems of routing goods from depots to consumers or between different logistical sites are very important. Drawing adequate plans may produce significant savings for many distribution systems. Therefore, the Vehicle Routing Problem (VRP) is a well-known problem studied in Operational Research. It deals with finding the optimal delivery routes configuration from one or several depots to a number of customers using a fleet of capacitated vehicles, while satisfying some constraints. The solution of a VRP is a set of minimum cost routes which fulfill the customers’ requirements. In the classical version of the problem, only one depot is considered with an unlimited fleet of vehicles where each vehicle performs exactly one circuit. Several operational constraints can be considered in more practical applications of the VRP.

In this research, we focus on two variants of the VRP: the Profitable VRP and the VRP with Multiple Trips, which we combine to obtain a new variant not addressed in the literature to our knowledge and involving a new challenge in term of solving approaches. This problem which can be named the Profitable Vehicle Routing Problem with Multiple Trips (PVRPMT) calls for the determination of a set of routes for a given heterogeneous set of vehicles visiting a selective subset of customers such that: i) it is difficult, if not impossible, to visit all customers for a lack of logistical means or for insufficiency of the offer; ii) each visited customer provides a fixed profit that is recognized in the objective function to maximize. The latter is the difference between the total profit generated by all visits and the resulting total transportation cost; iii) each vehicle may perform several routes in the same planning period but does not exceed a strict duration limit (temporal capacity); iv) for each route, the total load does not exceed the vehicle capacity (physical capacity). The motivation to study this problem come not only from its theoretical interest but also from its practical relevance. For example, the size of the vehicle fleet indicates the capital invested in logistics resources. That the reduction of this size is naturally more
desirable, the problem counts as a strategic choice in the problem reading to guide the optimization towards reducing both capital and operating (scheduling) costs. In this situation, there might be a trade-off between higher scheduling costs and lower vehicle ownership-based costs. Otherwise, this problem is very practical where daily drivers’ schedules must be achieved with a limited vehicle fleet and short-distance distribution networks. Furthermore, when we are unable to respond to all customer requests, the problem reading should be different. A subtle selection of customers to be served is required. It should be based on a clear and measurable criterion. It is therefore important to define a significantly manner to classify customers according to their importance, behavior and costs implications, and to translate this into a global criterion that can be optimized as the total cost associated with the transportation program. This issue of profitability will be raised when the instances are to be conceived and tested, because most companies do not have a carefully designed evaluation of each customer’s profit. The latter should not only be related to the volume of its current request but must take into account other considerations, related to the future potentials of this customer. And the value must also be reasonably projected on a coherent scale with the transportation costs that appear in the objective function in order to ensure significant aggregation-based assessments for the computed solutions.

The addressed problem concerns the PVRPMT as a new variant of practical VRP. Note that this kind of problems would be attractive in particular for planning mechanisms of ERP software which must integrate a miscellaneous range of situations of management and control. For a more devoted emphasis mainly to the operational context, our objective is multiple with this problem combining at hand: the maximization of the total collected profit, the minimization of the total routing costs and indirectly the maximization of the use of resources (vehicles and drivers). The issue of different mathematical models for this problem is to be addressed and used to seek solution optimality for small-size instances. Giving the complexity of the problem which is NP-hard, two constructive polynomial heuristics with two different greedy-based constructions will be proposed as a first approach to solve the large-size instances. Their effectiveness is tested and then enhanced, in a second solving approach, with an iterative ameliorative procedure using successive local searches with multiple and sequential neighborhood structures. For instances generation, we point out a new and quick discursive approach for profits quantification which is more significant than those existing in the literature.

The remainder of this paper is organized as follows. Section 2 reviews the related research literature. In section 3, we formally present the optimization problem PVRPMT with a detailed theoretical graph description and we introduce the corresponding proprieties and notations. Furthermore, we determine this problem complexity. In Section 4, based on two strategies of sub-tours elimination constraints, we provide four mathematical models of the problem including MILP and 0-1 ILP. Section 5 describes two constructive heuristics and an enhancement procedure based on the Variable neighborhood Descent algorithm (VND). The computational experiment is presented in Section 6 to obtain insights about the different performances of the optimizing proposed algorithms. This section begins with a new precision about the evaluation of profitability indices to be incorporated into the numerical instances’ files and subsequently into the computation of the objective functions in adequacy with the measures adopted for the transportation costs. We draw conclusions and discuss future research directions in Section 7.

2. Literature Review

The problem we focus on is a combination of the two known variants such that: the vehicle routing problem with multiple trips or with multiple uses of vehicles (noted by VRPMT or VRPM) and the profitable vehicle routing problem (PVRP). To the best of our knowledge, these problems are studied separately in literature, excepting some special recent works[2, 3, 5], but with time windows considerations. For the latter, the concept of Multiple Trips is obtained indirectly by imposing a duration limit on each tour and not on the workday duration. Thus, the problem may look like a Capacitated VRP, where the multiple uses of vehicle is a consequence since the tours duration limit is fixed sufficiently small. So, the tours could be seen as parallel tours conducted by several vehicles. However, the integration of time windows constraint into the same problem gives the importance to the tours’ order and consequently gives a special (restricted) Multiple Trips consideration. In this section, we devote two sub-sections to review the literature of the VRPMT and the PVRP.

2.1. The Vehicle Routing Problem with Multiple Trips

The VRPMT is an extension of the classical VRP in which the same vehicle may perform several routes in the same planning period (workday). In this case, the fleet includes a fixed number of available vehicles. However, the duration of a vehicle workday which is made of a set of successive routes does not exceed a certain limit.

This problem has a very practical importance. For example, in the home delivery of perishable goods like food, routes are of short duration and must be combined to form a complete workday[1]. Some real-world cases dealing with this kind of problem appear in applicative research publications. For instances, note the paper of Brandao et al. 1997[6] which studies the VRP in Burton’s Biscuit Ltd; and the case study of the logistical activities of Santa Fe Indonesia (precisely in the office of Jakarta, a company specialized on relocation services to individuals as well as companies) which is considered in[7]. Derigs et al. 2011[8] study some real cases dealing with this problem from a consultancy company in the air cargo transportation field.
(air cargo road feeder services). Planned and ad-hoc tasks, different resources and environment constraints are taken into account, such as driving time with breaks, working time with minimum requirements in relation to the health and the safety of persons performing mobile road transport activities. These problems deals with different applications, such as: the transportation of live animals to a slaughterhouse, the transportation of the meals or the flight food from a central kitchen to the aircraft, the transportation of goods from suppliers to retailers using one or more cross-docks. More additional constraints, such as time windows, split delivery, heterogeneous fleet, are considered respectively in[9, 10, 11and 12]. Grüner et al. 1999[13] develop a decision support system (DSS) which is designed in order to assist planners of the German postal service at the Deutsche Post AG. The main goal is to reduce the costs of the letter-mail transportation. In[14], a vehicle routing problem including multiple trips in a health care organization which operates about 240 buildings of medical offices in Southern California is treated. Finally, we mention Hernandez et al. 2009[15] which present a VRPMT applied in a context of optimizing the agronomic performance and its impact on the environment.

In the academic literature, the papers treating the VRPMT consider in general additional constraints mainly the time windows. Azi et al. 2007[1] study this problem with time windows consideration and a single vehicle’s use. The extension for multi-vehicle version is tackled by Azi et al.[2, 3, 4] and Macedo et al. 2011[5]. Other works include also both multiple trips and time windows such as[6, 7, 8, 11, 12, 14, 15, 16, 17, 18, 19 and 20]. The overtime constraint is incorporated to the VRPMT in[14, 16, 21, 22, 23 and 24]. Heterogeneous fleet is considered with this problem in[7, 10, 12, 16 and 27]. Some VRPMT are studied by including a products compatibility constraint which forbidden to gather in a same vehicle route two or more different categories of goods[7, 8, 16, 24, and 27]. The multiple trips constraint is used in VRP with backhaul in[7 and 12], combined with allowed split delivery case in[12], with location problem in[25 and 26], with a planning horizon of several days in[14 and 27], with multiple depots in[28], with meal breaks during the driving time in[11]. The problem is studied in a dynamic environment in[4 and 8].

In term of solving methods, few papers develop exact approaches for the VRPMT, because of the problem’s complexity, such as[1, 2, 5, 15 and 17] using MILPs, column generation with branch-and-price algorithm and network flow-based models. Note that solving the VRPMT may consist on solving a routing and packing problem. In this level, different heuristics and metaheuristics are developed. For the routing phase, we can distinguish some constructive heuristics used to obtain an initial solution, such insertion heuristics used in[3, 4, 10, 14, 20, 22, 27 and 29], the Clark and Wight algorithm[24, 25 and 31], the modified Clark and Wight algorithm[25 and 31], the nearest neighborhood method[25 and 29], clustering algorithms[7, 19 and 24], the sweep-based algorithm[20, 21, 28 and 32], set covering-based approaches[7 and 32] and the petal method[32]. The initial obtained solutions are enhanced by different methods: we find the tabu search in[6, 13, 14, 18, 19, 20, 21, 23, 25, 27, 28 and 29], the simulated annealing in[3, 25, 26 and 31], neighborhood search based on insertion moves in[10, 18, 22, 24, 25, 29 and 31], on swap moves in[22, 24, 25, 26, 29 and 31], on 2-opt in[18 and 25], on adaptive memory in[21 and 30]. Other specific neighborhood search algorithms are proposed such as large neighborhood search in[3 and 4], guided neighborhood search in[8]. In[8], we find also a decomposition approach. In[18], a filter and fan procedure is developed. For the packing phase, some well-known heuristics for bin packing problems are used, specially the best fit decreasing in[16, 22, 24, 25 and 30]. In[8], a greedy packing procedure is used and a fuzzy theory-based method is proposed in[33].

The VRP with multiple trips, but no other additional constraints, is addressed through heuristics in Brandao and Mercier 1998[7], Taillard et al. 1996[23], Olivera and Viera 2007[21], Petch and Salhi 2004[22].

2.2. The Vehicle Routing Problem with Profit

VRP with profits are a generalization of the vehicle routing problems. Given a fixed-size fleet of vehicles, it might not be possible to serve all customers. Thus, a known profit is associated with each demand node and the customers must be chosen based on their associated profit minus the travelling cost to reach them in the solution. The idea to associate a known profit at each customer is made by Dell’Amicco et al. 1995[35]. The constraint to visit all the customers is relaxed, but for each unvisited customer a given penalty has to be paid. The objective function is to find a balance between these penalties and the cost of the tour. Profits are explicitly considered both in the Vehicle Routing Problem (VRP) and the Traveling Salesman Problem with Profits (TSPP), as stated by Feillet et al. 2005[34] in an excellent comprehensive survey. The TSPP can be formulated as a discrete bi-criteria optimization problem where the two goals are referred: maximizing the profit and minimizing the traveling cost. It is also possible to use one of the goals as the objective function and the other as a constraint. These problems can be divided into three categories according to Feillet et al. 2005[34]: a) the Profit Tour Problem (PTP), b) the Selective TSP (STSP), and c) the Prize–Collecting TSP (PCTSP).

For the first problem (PTP), the objective is to maximize the difference between the total collected profit and the traveling cost. Feillet et al. 2005[34] survey lists various modeling approaches to TSPP and exact methods as well as heuristic solution methods. Archetti et al. 2009[36] study the capacitated version of the PTP and propose exact and heuristic procedures for it. More recently, the authors[37] develop an exact approach based on a branch-and-price algorithm. A restricted master heuristic is applied at each node of the branch-and-bound tree in order to obtain primal bound values.

For the second problem which is known through three
equivalent names in the literature: the Orienteering Problem (OP), the Selective TSP (STSP), or the Maximum Collection Problem (MCP), the objective is to maximize the collected profit such that the total traveling cost or distance does not exceed a certain upper bound. The multi-vehicles version of the OP is very similar to the VRP, except that vehicles are assumed to be with unlimited capacity and there is a time constraint on each tour. This problem is called the Team Orienteering Problem (TOP). Recently, Vansteenwegen et al. 2011[43] provides a survey of the existing OP and TOP. In this paper, the applications and the solving approaches about the two problems is reviewed listing various modeling approaches and exact as well as heuristic solution methods. In addition, many relevant variants of these problems are formally presented.

For the third problem which is named the Prize–Collecting TSP or also known as the Quota TSP concerns the determination of a tour with the minimum total traveling cost where the collected profit or prize is greater than a lower bound. Note that the PVRP is also a kind of PTP when considering comparing to TSPP that vehicles have physical loading capacities. In[38], The PVRP is applied in the reverse logistics where a firm aims to collect cores from its dealers. The problem is an extension of the classical multi-depots vehicle routing problem (MDVRP) in which each visit to a dealer is associated with a gross profit and an acquisition price to be paid to take the cores back. First, two mixed-integer linear programming (MILP) are presented. Then, a Tabu Search-based heuristic is proposed to solve medium and large-sized instances. In[39], the research deals with a VRP in which the total profit is to be maximized subject to market competition. The PTP in a dynamic environment is considered in[40]. In this problem, the rewards (profits) are unknown for the customers which are not yet served. Indeed, the rewards depend on competitors’ prices and auctions. In another extension of the problem, the profit is associated to each edge non to vertices. The problem is then called the Profitable Arc Routing Problem (PARP). Archetti et al. 2010[41] study the capacitated undirected version (UCARP) and develop a branch-and-price algorithm, several heuristics based on Variable Neighborhood Search (VNS) and two Tabu Search heuristics. Zachariadis et al. 2011[42] propose another local search approach for the UCARP. Two solution neighborhoods are considered and the overall search is coordinated by the use of the promises concept.

The problem studied in this paper is a new extension which combines the two previous variants.

3. Problem Definition and Notation

3.1. Problem Description and Notation

We consider a complete undirected graph $G = (V, E)$, where $V = \{0, \ldots, n\}$ is a set of vertices and $E$ is a set of edges. Vertex 0 represents the depot and a fleet of vehicles $k = \{1, \ldots, m\}$ is based. Each vehicle has a limited capacity $Q$ (or $Q_k$ if with heterogeneous fleet) and a maximum number of trips $L$. Note that, the parameter $L$ is defined to facilitate the problem formulation and can be used as a real constraint, i.e. as a limit to be respected. However, the best value of $L$ can be experimentally identified. An edge $(i,j) \in E$ represents the possibility to travel from customer $i$ to customer $j$. A non-negative demand $q_i$, profit $p_i$, and time service $S_i$, are associated with each customer $i$ (with setting $p_0 = q_0 = 0$). A travel time $t_{ij}$ and cost $c_{ij}$ are associated with each edge $(i,j) \in E$. Each vehicle starts and ends its tour at vertex 0, and can visit any subset of customers with a total demand that does not exceed the used vehicle capacity $Q_k$. In addition, there exists a time horizon denoted by the duration limit $T_{\text{max}}$ which establishes the duration of a workday. It is assumed that all parameters are nonnegative integers and the environment is deterministic.

This problem named the PVRPMT consists on determining a set of routes and to assign each route to one vehicle, such that the same vehicle can be used by several routes while respecting the time horizon. The objective is to maximize the difference between the total collected profit and the cost of the total traveled distance. Note that the following properties:

- The optimal solution may be composed by a subset of customers.
- Each route starts and ends at the depot,
- The total customers’ demand in the same route does not exceed the physical capacity of used vehicle,
- The duration of routes assigned to the same vehicle does not exceed $T_{\text{max}}$.
- The profit associated at each customer is fixed and can be collected by any vehicle.

3.2. Problem Complexity

Olivera and Viera[21] proved that the VRPMPT is NP-hard as well as the PVRP[34]. The studied problem represents the combination of these two NP-hard problems. This makes it also NP-hard. In addition, PVRPMT is a generalization of the classical VRP. There are instances of PVRPMT in which there exist enough vehicles in the fleet able to optimally visit during the workday all customers using one vehicle for each tour. These cases are naturally reduced to a classical VRP. As the VRP is an NP-hard[22], the PVRPMT is also NP-hard.

4. Mathematical Models for the PVRPMT

The design of the VRP solution stands against the presence of the sub-tours. For that, different sub-tours elimination constraints are proposed. The most classical constraint and the most used in the literature can be written in this way:
\[
\sum_{i \in S} \sum_{j \not\in S} x_{ij}^k \leq |S| - 1 \quad \text{with} \quad S \subseteq \{1, \ldots, n\}; 2 \leq |S| \leq (n-1)(1)
\]

This formulation is exponential and influences strongly the resolution time.

In this section, we propose four mathematical formulations for the PVRPMT. The difference between them is based on the strategy used to eliminate the sub-tours and the decision variables definitions. The different indices, parameters and decision variables are given in Table 1.

<table>
<thead>
<tr>
<th>Table 1. Nomenclature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indices</td>
</tr>
<tr>
<td>(i, j) : customer index</td>
</tr>
<tr>
<td>(l) : trip index</td>
</tr>
<tr>
<td>Variables</td>
</tr>
<tr>
<td>(x_{ij}^k) : \begin{cases} 1 \text{ if edge } (i, j) \text{ is assigned to the } t^{th} \text{ trip of vehicle } k \ 0 \text{ Otherwise} \end{cases}</td>
</tr>
<tr>
<td>(\delta^k) : \begin{cases} 1 \text{ if the vehicle } k \text{ is used during the trip } l \ 0 \text{ Otherwise} \end{cases}</td>
</tr>
<tr>
<td>(y_{il}^k) : \begin{cases} 1 \text{ if } i \text{ is the } t^{th} \text{ visited customer in the trip } l \ 0 \text{ Otherwise} \end{cases}</td>
</tr>
<tr>
<td>(U_i) : variable associated to customer (i) used to reformulate the sub-tour elimination constraints</td>
</tr>
<tr>
<td>Parameters</td>
</tr>
<tr>
<td>(c_{ij}) : cost associated with the edge ((i, j))</td>
</tr>
<tr>
<td>(t_{ij}) : time to traverse edge ((i, j))</td>
</tr>
<tr>
<td>(Q) : capacity of the vehicle</td>
</tr>
<tr>
<td>(m) : number of available vehicles</td>
</tr>
<tr>
<td>(L) : maximum number of trips that can be made by one vehicle</td>
</tr>
<tr>
<td>(n) : number of vertices</td>
</tr>
<tr>
<td>(T_{max}) : working day time limitation</td>
</tr>
<tr>
<td>(q_i) : demand of customer (i)</td>
</tr>
<tr>
<td>(p_i) : profit collected at customer (i)</td>
</tr>
<tr>
<td>(S_i) : service time at customer (i)</td>
</tr>
<tr>
<td>(M) : a big positive number</td>
</tr>
</tbody>
</table>

4.1. Modeling with 0-1 Integer Linear Programming

We start with the idea to specify for each assigned customer his order in the trip. This idea has been applied in the scheduling problem. For each job, we determine the position of the job in the sequence. We use the visit order of customer \(i\) in the trip in order to eliminate the invalid tours. Accordingly, we eliminate the subset \(S\) and the associated constraints. We consider the following binary decision variables:

\[
x_{ij}^k = \begin{cases} 1 \text{ if edge } (i, j) \text{ is assigned to the vehicle } k \text{ during the trip } l \\ 0 \text{ Otherwise} \end{cases}
\]

\[
y_{il}^k = \begin{cases} 1 \text{ if } i \text{ is the } t^{th} \text{ visited customer in the trip } l \\ 0 \text{ Otherwise} \end{cases}
\]

A new decision variable \(y_{il}^k\) is added and informs about the visit order of customer \(i\) in the trip \(l\).

Based on the choice of the principal variable in the formulation, we can distinguish two different mathematical models.

- First formulation: 0-1ILP1 with \(x_{ij}^k\) as principal variables.

Here, the second variable \(y_{il}^k\) is used just to make the connection between the edge and the vertices. The resulting formulation is the following:

\[
\text{Max } \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=0}^{m} \sum_{l=1}^{L} (p_i - c_{ij})x_{ij}^k
\]

Subject to

\[
\sum_{j=0}^{n} x_{ij}^k = \sum_{r=0}^{n+1} x_{ir}^k \quad \forall i = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
\sum_{r=0}^{n+1} \sum_{k=0}^{m} \sum_{l=1}^{L} x_{ir}^k \leq 1 \quad \forall i = 1, \ldots, n
\]

\[
\sum_{r=0}^{n+1} x_{ir}^k \leq 1 \quad \forall i = 1, \ldots, m; l = 1, \ldots, L
\]

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} q_i x_{ij}^k \leq Q \quad \forall k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{l=1}^{L} (t_{ij} + S_i) x_{ij}^k \leq T_{max} \quad \forall k = 1, \ldots, m
\]

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{k=0}^{m} x_{ij}^k - \sum_{j=0}^{n} x_{hj}^k = 0 \quad \forall h = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
x_{ij}^k + x_{it}^k \leq 1 + y_{ij}^k \quad \forall i, j, t = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
y_{ij}^k = x_{ij}^k \quad \forall j = 1, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
x_{ij}^k + x_{jt}^k \leq 2 - \sum_{t=0}^{n+1} y_{ij}^k \quad \forall i, j, t = 1, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
M \sum_{i=0}^{n} \sum_{j=0}^{n} x_{ij}^k \geq \sum_{i=0}^{n} \sum_{j=0}^{n} x_{ij}^k \quad \forall k = 1, \ldots, m; l = 1, \ldots, L
\]

\[
x_{ij}^k y_{il}^k \in \{0, 1\} \quad \forall i, j, t = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L
\]

In this formulation, the objective is to maximize the overall collected profit minus the transportation cost. The constraints (3-5) guarantee that each customer is visited at most once. In (6), if the route \(l\) exists, it should start and finish in the depot. (7) represents the capacity constraint. The limit duration on a workday is restricted by (8). (9) represents the flow conservation constraint. The constraint (10) establish the relation between the edge \((i,j)\) and the relevant position of \(i\) and \(j\). (11) represents the initialization of the counter and (12) stands against the addition of a customer if the route is closed. In (13), each constructed trip should start in the depot. (14) represent the integrity constraints.

- Second formulation: 0-1 ILP2 with \(y_{il}^k\) as principal variables.
variables
In this model, we conserve the constraints which establish the relation between \( x_{ij}^k \) and \( y_{ij}^k \) such as ILP1. When it is possible, we integrate the variable \( y_{ij}^k \) in the objective function and in the constraints. The objective is to test the influence of this transformation on the upper bound and to know which expression can conserve much more information if the integrity constraints are relaxed. The resulting model is as follow:

\[
\begin{align*}
\text{Max} & \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{L} p_{i,j} x_{ij}^{kl} - \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=1}^{m} \sum_{l=1}^{L} c_{ij} x_{ij}^{kl} \\
\text{subject to:} & \quad \sum_{j=0}^{n} x_{ij}^{kl} = \sum_{i=1}^{n} x_{ij}^{kl} \quad \forall i = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L \\
& \quad \sum_{i=0}^{n} \sum_{j=1}^{n} \sum_{k=1}^{m} \sum_{l=1}^{L} y_{ij}^{kl} \leq 1 \quad \forall i = 1, \ldots, n \\
& \quad \sum_{j=1}^{n} y_{ij}^{kl} \leq 1 \quad \forall t = 1, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L \\
& \quad \sum_{i=0}^{n} y_{ij}^{kl} \leq 1 \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \\
& \quad \sum_{i=0}^{n} \sum_{j=1}^{n} q_{ij} x_{ij}^{kl} \leq Q \quad \forall k = 1, \ldots, m; l = 1, \ldots, L
\end{align*}
\]

\(4.2. \text{ Modeling with Mixed Integer Linear Programming}\)

To overcome the limitation of the classical sub-tours elimination constraint, Miller et al. 1960[44] propose a new ones which are corrected by Kara et al. 2004[45]. In this level, we propose an adaptation of these constraints to our problem. For that, we remove the decision variable \( y_{ij}^k \) and we define \( U_i \) as an associated variable to customer \( i \) used to reformulate the sub-tours elimination constraints:

Firstly, we present the ordinary model. Then an extension of this model with some proposed cuts.

• Third formulation: MILP1 (without cuts)

The formulation is the following:

\[
\begin{align*}
\text{Max} & \quad \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=1}^{m} \sum_{l=1}^{L} (p_{i,j} - c_{ij}) x_{ij}^{kl} \\
\text{subject to} & \quad \sum_{j=0}^{m} x_{ij}^{kl} \leq 1 \quad \forall i = 1, \ldots, n \\
& \quad \sum_{i=0}^{n} q_{ij} x_{ij}^{kl} \leq Q \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \\
& \quad \sum_{i=0}^{n} \sum_{j=0}^{m} (t_{ij} + S_i) x_{ij}^{kl} \leq T_{\max} \quad \forall k = 1, \ldots, m
\end{align*}
\]
\[ \sum_{i=0}^{n} \sum_{j=0}^{n} x_{lh}^{kl} = 0 \quad \forall h = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L \] (40)

\[ 0 \leq U_i \leq Q \quad \forall i = 0, \ldots, n \] (41)

\[ U_i - U_j + \sum_{k=1}^{m} \sum_{l=1}^{L} (Q - q_i - q_j) x_{ij}^{kl} \leq Q - q_j \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \] (42)

\[ \delta^{kl} \leq \sum_{i=0}^{n} \sum_{j=0}^{n} x_{ij}^{kl} \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \] (43)

\[ \sum_{i=0}^{n} \sum_{j=0}^{n} x_{ij}^{kl} \leq M \delta^{kl} \quad k = 1, \ldots, m; l = 1, \ldots, L \] (44)

\[ x_{ij}^{kl} \in \{0, 1\} \quad \forall i \neq j = 0, \ldots, n; k = 1, \ldots, m; l = 1, \ldots, L \] (46)

\[ \delta^{kl} \in \{0, 1\} \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \] (47)

To integrate the new decision variable, three additional constraints which establish the relation between the two decision variables are adjoined. (43) means that if the route \(l\) exists, it should start and finish in the depot. (44) guarantees that if the edge \((i,j)\) is assigned to the trip \(l\), this latter should be constructed. The opposite case is presented by (45). This constraint prohibits the construction of empty route, i.e. if the route \(l\) is constructed, at least one edge must be assigned at this trip.

### 4.3. The Heterogeneous Fleet Case

In this subsection, we proposed a formulation extension to solve the special case of PVRMT with heterogeneous fleet. So, each vehicle \(k\) has its own capacity \(Q_k\). Then, some modifications on MILP2 are done. The constraint (4.3) is replaced by (5.3). The constraints (4.6) and (4.7) are replaced respectively by (5.6) and (5.7):

\[ \sum_{i=0}^{n} \sum_{j=0}^{n} q_i x_{ij}^{kl} \leq Q_k \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \] (48)

\[ 0 \leq U_i \leq Q_k \quad \forall i = 0, \ldots, n \] (49)

\[ U_i - U_j + \sum_{k=1}^{m} \sum_{l=1}^{L} (Q_k - q_i - q_j) x_{ij}^{kl} \leq Q_k - q_j \quad \forall k = 1, \ldots, m; l = 1, \ldots, L \] (50)

### 5. Solving Algorithms

The present section is devoted to the description of the main steps of the implemented algorithms. We start by explaining the solution coding scheme. Then, constructive heuristics used to build initial feasible solutions to the problem are described. Finally, improvement procedures based on Hill Climbing and Variable Neighborhood Descent (VND) are proposed.

#### 5.1. Solution Coding Scheme

To further speed up the computation, we use tree-array data structure to represent a solution. The information concerning the customer is stored into the first array \((V1[j])\). So, its size is equal to the number of customers \(n\) (we eliminate the depot from the solution representation). First, the visited customers are stored adjacently in the order of visit. Then, the unvisited customers are inserted in the end of the array. It is important to know the start and the end of each trip. This information is done by the second array \((V2[j])\). This latter is a binary vector. Finally, the third array \((V3[j])\) informs about the index of the used vehicle to visit the correspondent customer. Using this structure, the lecture of the solution is clear and easy, and changes can be performed very quickly and in a constant time. The solution is coded as follows:

Let the matrix \(V[i][j]\) with \(i \in \{1, 2, 3\}\) and \(j \in \{1, 2, \ldots, n\}\) be the following:

![Figure 1. Illustrative Example](image_url)
Table 2. Coding Solution of the illustrative example

<table>
<thead>
<tr>
<th>j</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1[j]</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>7</td>
<td>12</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>V2[j]</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>V3[j]</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

5.2. Constructive Heuristics for the PVRPMT

The goal of this subsection is to construct good feasible solutions for the problem. We propose two greedy constructive heuristics. These heuristics use some local optimalities in certain steps of the algorithm.

5.2.1. Heuristic H1

The insertion heuristic is used to build an initial feasible solution. In every iteration, the procedure evaluates all feasible insertions of unvisited nodes and selects the node representing the best insertion. An insertion is evaluated with the following criterion. Let \( i \) be some node in a tour and let \( j \) be a node candidate for the insertion. Let \( Chg_{act} \) and \( Tm_{act} \) denote the actual charge and the traveled time for the current used vehicle respectively. The insertion of \( j \) is feasible if the capacity constraint and the constraint of time duration are verified (i.e. (51) and (52) are verified):

\[
Chg_{act} + q_j \leq Q_k \quad (51)
\]

\[
Tm_{act} + t_{ij} + S_j + t_{j0} \leq T_{max} \quad (52)
\]

Then, the best insertion is determined by the pair \((i, j)\) for which \( \pi_{ij} \) is maximum:

\[
\pi_{ij} = p_j - c_{ij} \quad (53)
\]

Where \( P_{ij} \) is the profit generated by the insertion of the customer \( j \) after the customer \( i \), \( p_j \) is the profit associated with customer \( j \) and \( c_{ij} \) is the cost of transportation between \( i \) and \( j \). The route is considered as completed if the vehicle can not receive any more other customers due to its physical capacity. The vehicle returns to the depot and a new trip starts with the same used vehicle in the last iteration. The vehicle is changed only if the daily time horizon \( T_{max} \) will be violated by the addition of a new trip with the same vehicle. In this case, the next trip is done by the next vehicle (see Figure 2).

Remark 1: in the case of a heterogeneous fleet, the vehicles are sorted in the decreasing order of their physical capacities. This order is used as a priority rule for the vehicle choices.

5.2.2. Heuristic H2

This heuristic H2 is almost identical with heuristic H1 with the following basic difference: to construct the trip, we use the same previous procedure, but the used vehicle is not necessary kept in prior for the next trip. At each iteration, we choose the vehicle which has the longest remaining time service by breaking ties with the largest capacity order (see Figure 3).

Figure 2. Constructive Heuristic 1
combining the three last algorithms in a Variable Neighborhood Descent procedure.

5.3.1. Hill Climbing Procedure

The local search algorithms show its performance to solve various variants of routing problems. Here we use the Hill Climbing heuristic which belongs to the family of local search methods which often built on neighborhood moves that make small changes to the current solution.

The insert move consists of removing one customer from a current position \( j \) (origin position) and putting him into another new position \( k \). The destination route (i.e. the route that contains the new position) can be an existing route or a new one. To take into account all the possible moves, a new decision variable \( \alpha \) is defined:

\[
\alpha = \begin{cases} 
\text{prec : } j & \text{will be inserted in the last trip} \\
\text{suv : } j & \text{will be inserted in the next trip}
\end{cases}
\]

All depends on the positions of \( j \) and \( k \) and the variable \( \delta \).

* If \( j = k \), three scenarios can be distinguished:
  - \( k \) is the position of the first visited customer in the current trip (i.e. \( V2[k-1] = 1 \))
    - if \( \alpha = \text{prec} \) the customer \( i \) will be inserted into the last trip (i.e. \( V2[k] = 0, V2[k] = 1 \) and \( V3[k] = V2[k-1] \))
    - if \( \alpha = \text{suv} \) no change in the initial solution
  - \( k \) is the last visited customer in the current trip (\( V2[k] = 1 \))
    - if \( \alpha = \text{prec} \) no change in the initial solution
    - if \( \alpha = \text{suv} \) the customer \( i \) will be inserted into the next trip (i.e. \( V2[k] = 1, V2[k] = 0 \) and \( V3[k] = V3[k+1] \)).
  - Otherwise: there is no change

* If \( j \neq k \), according to the value of \( V2[k] \) two different cases can appear:
  - If, in the initial solution, \( V2[k] = 1 \): two insert moves are presented
    - \( \alpha = \text{prec} \): \( i \) will be the last visited customer who belongs to the current trip (\( V2[i] = 1 \) and \( V3[i] = V3[k] \))
    - \( \alpha = \text{suv} \): \( i \) will be the first visited customer in the next trip (\( V2[i] = 0 \) and \( V3[i] = V3[k+1] \)).
  - if \( V2[k] = 0 \), the new solution will be presented as follows: \( V1[k] = V1[j] \), \( V2[k] = 0 \) and \( V3[k] = V3[k-1] \).

The swap move consists on exchanging two customers. In the new solution just the position of the two selected customers will be exchanged (\( V1[k] = V1[j] \)).

The cross-exchange operator consists on interchanging non-consecutive customer segments between the same or two different routes with the restriction that the orientation of them be maintained.

Remarq 3: Note that, to eliminate the neglected move, for the two first neighborhoods (insertion and swap), at least one from the two selected customers should represent a visited customer in the initial solution. For the cross-exchange operator, the two selected customers should be visited in the initial solution.

5.3.2. Variable Neighborhood Descent Algorithm

The second idea is to test the influence of combining the
three neighborhood structures in the same algorithm. So, we
develop a variable neighborhood decent algorithm (VND).
The VND is the simplest variant of the variable
neighborhood search heuristic (VNS) which performs
several descents with different neighborhoods until a local
optimum for all considered neighborhoods is reached. Let
\(N_1, N_2, \ldots, N_k\) denote a set of \(K\) neighborhood structures
(i.e., \(N_i(S)\) contains the solution that can be obtained by
performing a local change on \(s\) according to the \(i^{th}\) type).
The VND works as follows:

**Step 1.** Choose an initial solution \(s\) in \(S\).

**Step 2.** Set \(i := 1\) and \(s_{best} := s\).

**Step 3.** Perform a descent form \(s\), using neighborhood
\(N_i\). Let \(s'\) be the resulting solution. If \(f(s') < f(s)\) then
set \(s := s'\). Set \(i := i + 1\). If \(i \leq K\) then repeat Step 3.

**Step 4.** If \(f(s) < f(s_{best})\) then go to Step 2; else stop.

### 6. Computational Results

In the following, obtained results concerning the
performance of the different proposed methods are reported.
First, the test instances construction, the parameters choice
and the profit quantification are presented. Then, the
performance of the equivalent presented models are tested and compared. Finally, the last subsection is devoted to the heuristics procedures performance (constructive, Hill Climbing, VND). The MILPs and solving algorithms was tested on a Intel Core 2 Duo CPU 2.20 GHZ and 4.00 Go RAM. The codes are written in C++, using CPLEX libraries
for the first part. All the algorithms were stopped before a
computational time of one hour at most.

#### 6.1. Test Instances and Parameters Choice

The tests will be applied first on our own benchmark
devoted for small-size instances and then on the benchmarks of
Taillard et al. 1996[23] taken from the VRP library with adaptations and some extended data. In our benchmark, twenty small-size instances are generated randomly. A certain setting is used to obtain interesting instance values
according to the practice and real-case situations. For each instance, we indicate the number of vertices \(n\) which ranges from 6 to 20. The customers are randomly distributed in
two-dimension area, and the depot is set at point (0, 0). the
fleet of vehicles \(m\) accounts for 2 or 3 vehicles. The vehicle
physical capacity limit \(Q\) ranges from 1000 to 3000 kgs. The horizon time limit \(T_{max}\) for all the small size instances is
equal to 480 minutes (i.e. 8 hours of workday).

Note that in the literature according to[36], the profit \(p_i\)
of customer \(i\) depends on the three parameters namely: cons, \(h\) and the customer’s demand \(q_i\), where \(h\) is a random ratio
number uniformly generated in the interval\([0,1]\) and cons is a constant factor that measures the profit according to a sale turnover. In[36], the authors consider \(p_i = (con + hq_i)\) and suppose that \(cons=0.5\) indifferently with the level of
greatness of the other values used for the instances data.

This implies a lack of guarantee of a necessary coherence between
the different values while the choices of the units and the
level of greatness of the four used data are unambiguous:
customers’ demands \(q_i\), distances \(d_{ij}\), transportation times \(t_{ij}\) and transportation costs \(c_{ij}\).

In our opinion, it is important to obtain meaningful and
significant proportions of the profits according to the
transportation costs. In our case, they are assimilated to \(d_{ij}\). Concerning the profit calculation, we take as reference a
general realistic model where the logistical cost represents
between 5 and 10 per cent of the sale turnover, and the gross
benefit may represent between 20 and 50 per cent of the
sale turnover. So, it is possible to have a global profit which
ranges almost between 3 and 5 times the gobal transportation cost.
The distribution of this global profit on customers may
respect the demand quantity \(q_i\) of each customer \(i\). Let \(q_{max}\)
and \(q_{min}\) be respectively the greatest and the smallest value of
all demand quantities. \(q_{max}\) and \(q_{min}\) can be associated respectively to \(\alpha_{max}=5\) and \(\alpha_{min}=3\) factors. We use a projection
to calculate the factor \(\alpha_{i}\) of the customer \(i\) according to his
demand \(q_i\).

In order to estimate the transportation cost, we calculate
firstly the average distance, identify and \(a_i\) and \(p_i\), such as:

\[
\bar{d} = \frac{\sum \sum d_{ij}}{(n-1)^2}
\]

\[
\alpha_{i} = \frac{\alpha_{max} - \alpha_{min}}{q_{max} - q_{min}} \quad q_{i} - q_{min} = q_{max} - q_{min}
\]

\[
p_i = \alpha_{i} (1 + h) \bar{d}.
\]

Note that \(t_{ij}\) are also assimilated to \(d_{ij}\) and the level of
greatness of the vertices coordinates values is chosen
adequately to obtain significant \(t_{ij}\) values suitable with \(T_{max}\).

In addition, we point out that \(L\), which is used in MILP
formulations, represents the total number of trips which can
be made by a vehicle during \(T_{max}\). Its value can be estimated as :

\[
\min \left\{ \frac{\sum Q_k}{\min Q_k}, \frac{\sum Q_k}{\min Q_k} \right\}
\]

For the original instances of Taillard et al. 1996, we add all
previous adaptations and extensions, as well as the service
time \(S_t\) and the maximum number of trip which can be
assigned at each vehicle \(L\).

#### 6.2. Mathematical Models Experimentations

In order to test the proposed models, we used the
commercial solver CPLEX 10.0. of ILOG ®.

First, the four mathematical models are tested on the small
size instances and the results are reported in Table 3. The
optimal solution obtained by CPLEX is indicated under
column \(f\). The “-” means that we cannot obtain the optimal
solution before the time limit. The column \(UB\) represents
the solution of the linear relaxed problem (the upper bound).
In addition, \(\%\) represents the gap between the optimal
solution and the relevant upper bound. \( \% = (f UB) / UB \times 100 \).

We observe on Table 3 that the optimal value can be obtained within the time limit just for small size instances for the four mathematical models. For the instances where the number of vertices is superior to 17, the optimal solution cannot be determined. We can show also that computational time grow strongly with the increase of instance size. MILP 2 is able to solve more instances than the other proposed models and it clearly outperforms them in terms of computing time.

In Table 4, we try to compare the performance of the four mathematical models by confronting the obtained upper bounds. We choose as reference the upper bound of the first mathematical model (UB1) and we calculate the deviation of the different upper bounds in comparison with the first. \( (\%) = (UB w - UB1) / UB1 \times 100 \).

Different remarks can be taken into account. First by comparing the two strategies, the 0-1 ILP and the MILP, it is very clear that the upper bounds obtained by the mixed integer programming are better than those obtained by the 0-1 integer programming for all the tested instances. For the 0-1ILP, the choice of the principal variable has not great influences on the upper bounds quality (for the majority of the tested instances the upper bound obtained by the two models are equal) but strongly affects the number of iterations. For the MILP, the upper bounds of the mathematical models with cuts are better than those without cuts. That shows the efficiency of these cuts. MILP2 is able to solve more instances than the other mathematical models and outperforms them in terms of computing time and the upper bound quality. So, we can judge that the additional constraints represent valid cuts for our model.

### 6.3. Heuristics Procedure Performance

To test and compare the performance of our heuristics, we compute the obtained gaps of the obtained solutions comparing to the linear upper bounds. These gaps are given: first for the two constructive heuristics bounds, second for the three Hill Climbing based solutions, and finally for enhancement by using the Variable Neighborhood Descent algorithm. Gaps are also computed comparing to the optimal solution for the small-size instances. Then, to know the contribution generated by the VND algorithm, we calculate the gap between the initial solution given by the constructive heuristics and the lower bound. As it is previously mentioned, MILP2 represents the best formulation of our problem. So, the lower bound given by our heuristic will be compared with the upper bound obtained with this formulation.

The test includes our twenty instances generated randomly with adequate settings and forty adapted instances selected from the benchmarks of Taillard et al. 1996[23]. The results are shown in table 5 and 6.

The column \( f / UB \) represent respectively the optimal solution and the upper obtained by the linear relaxation of MILP2. The initial solution of our constructive algorithms (Constructive heuristic 1 and constructive heuristic 2) is given in column \( HCi \) and \( H2 \). The Hill Climbing improvement tested for the three neighborhood structures by the insertion move, the swap move and the cross-exchange are showed in the column \( Hci \), \( HCs \) and \( HCe \) respectively. The column \( VND \) gives the lower bound obtained with the Variable Neighborhoods Decent algorithm. The column \( gap \) represent respectively the gap between the lower bound (the \( VND \) solution) and the optimal solution and the upper upper respectively. The column \( gap^* \) calculates the enhancement generated by the \( VND \) algorithm by measuring the gap between the \( VND \) and the initial solution.

From the first observation and by comparing the three Hill Climbing shemes, we can see that these procedures give several acceptable results for the most tested instances. However, some of them still require further enhancement. On the 12 instances for which the optimal solution is determined, the \( VND \) algorithm is managed to find the optimal solution in 50% of cases. For the others, the gap is generally tiny, except for few exceptional instances, where it is quite significant reaching for the worst 33%. For the large size instances, this gap \( (gap^*) \) becomes important and grows continuously because it is computed compared to the linear bound and not to optimum. The \( VND \) algorithm enhances clearly the initial solution for the total of the tested instances. But, for few instances (i.e. instances 7 and 8) the improvement is small and the \( VND \) converges quickly to local maxima.

To crown all, we can conclude that the \( MILP2 \) (with cuts) represents the best formulation of our problem. The two constructive heuristics and the enhancement procedure based on the Hill Climbing and the \( VND \) algorithm produce acceptable solutions close to the optimal ones for small size instances. For the big size instances, the obtained solutions are the best till now. In a future works, we think about enhancing more and more the upper bounds with some polyhedral techniques to have a clearer idea about the performance of our proposed methods for the large size instances. To escape from a current local optimum, we should think to add, in a third solving phase, some perturbation moves to strengthen the search diversification in our algorithms. Best performances could be deduced in future works after some enhancements.
<table>
<thead>
<tr>
<th>Instance</th>
<th>0-ILP 1</th>
<th></th>
<th>0-ILP 2</th>
<th></th>
<th>MILP 1</th>
<th></th>
<th>MILP 2</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>f</td>
<td>CPU</td>
<td>UB1</td>
<td>%</td>
<td>f</td>
<td>CPU</td>
<td>UB2</td>
<td>%</td>
</tr>
<tr>
<td>Inst_6_2_2_1000</td>
<td>544.78</td>
<td>0.75</td>
<td>672.52</td>
<td>18.99</td>
<td>544.78</td>
<td>1.47</td>
<td>670.78</td>
<td>18.78</td>
</tr>
<tr>
<td>Inst_7_2_2_2000</td>
<td>844.77</td>
<td>6.84</td>
<td>979.69</td>
<td>13.77</td>
<td>844.77</td>
<td>6.42</td>
<td>979.69</td>
<td>13.77</td>
</tr>
<tr>
<td>Inst_7_2_3_3000</td>
<td>1780.4</td>
<td>19.2</td>
<td>2312.05</td>
<td>22.99</td>
<td>1780.43</td>
<td>11.13</td>
<td>2311.61</td>
<td>22.98</td>
</tr>
<tr>
<td>Inst_9_2_2_2000</td>
<td>607.73</td>
<td>760</td>
<td>700.05</td>
<td>13.19</td>
<td>607.73</td>
<td>180</td>
<td>699.575</td>
<td>13.13</td>
</tr>
<tr>
<td>Inst_9_2_3_2000</td>
<td>1932.6</td>
<td>259</td>
<td>2750.85</td>
<td>29.75</td>
<td>1932.59</td>
<td>608.92</td>
<td>2750.85</td>
<td>29.75</td>
</tr>
<tr>
<td>Inst_10_2_3_2000</td>
<td>2123.9</td>
<td>333</td>
<td>2553.53</td>
<td>16.82</td>
<td>2123.94</td>
<td>42.16</td>
<td>2472.93</td>
<td>14.11</td>
</tr>
<tr>
<td>Inst_11_2_2_2000</td>
<td>2310.9</td>
<td>3532</td>
<td>2969.87</td>
<td>22.19</td>
<td>2310.94</td>
<td>3032</td>
<td>2969.87</td>
<td>22.19</td>
</tr>
<tr>
<td>Inst_12_2_3_3000</td>
<td>_</td>
<td>_</td>
<td>2418.52</td>
<td>_</td>
<td>_</td>
<td>2418.52</td>
<td>_</td>
<td>2226.08</td>
</tr>
<tr>
<td>Inst_13_2_3_1000</td>
<td>_</td>
<td>_</td>
<td>3729.85</td>
<td>_</td>
<td>_</td>
<td>3729.85</td>
<td>_</td>
<td>3604.08</td>
</tr>
<tr>
<td>Inst_13_3_2_2000</td>
<td>_</td>
<td>_</td>
<td>3133.43</td>
<td>_</td>
<td>_</td>
<td>3133.44</td>
<td>_</td>
<td>3043.2</td>
</tr>
<tr>
<td>Inst_14_2_2_3000</td>
<td>_</td>
<td>_</td>
<td>4867.48</td>
<td>_</td>
<td>_</td>
<td>4867.48</td>
<td>_</td>
<td>4761.82</td>
</tr>
<tr>
<td>Inst_15_3_2_2000</td>
<td>_</td>
<td>_</td>
<td>5694.48</td>
<td>_</td>
<td>_</td>
<td>5694.49</td>
<td>_</td>
<td>5610.11</td>
</tr>
</tbody>
</table>
### Table 4. Comparison between the Upper Bounds

<table>
<thead>
<tr>
<th>Instance</th>
<th>(%)*_1</th>
<th>(%)*_2</th>
<th>(%)*_3</th>
<th>(%)*_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inst_6_2_2_1000</td>
<td>0</td>
<td>-0.259</td>
<td>-1.488</td>
<td>-5.164</td>
</tr>
<tr>
<td>Inst_7_2_2_2000</td>
<td>0</td>
<td>0.000</td>
<td>-2.256</td>
<td>-2.256</td>
</tr>
<tr>
<td>Inst_7_2_3_3000</td>
<td>0</td>
<td>-0.019</td>
<td>-0.511</td>
<td>-5.423</td>
</tr>
<tr>
<td>Inst_9_2_2_2000</td>
<td>0</td>
<td>-0.068</td>
<td>-1.564</td>
<td>-2.191</td>
</tr>
<tr>
<td>Inst_9_2_3_2000</td>
<td>0</td>
<td>0.000</td>
<td>-1.460</td>
<td>-1.481</td>
</tr>
<tr>
<td>Inst_10_2_3_2000</td>
<td>0</td>
<td>0.000</td>
<td>-3.156</td>
<td>-4.264</td>
</tr>
<tr>
<td>Inst_11_2_2_2000</td>
<td>0</td>
<td>0.000</td>
<td>-1.282</td>
<td>-1.426</td>
</tr>
<tr>
<td>Inst_12_2_3_3000</td>
<td>0</td>
<td>0.000</td>
<td>-3.372</td>
<td>-3.409</td>
</tr>
<tr>
<td>Inst_13_2_3_1000</td>
<td>0</td>
<td>0.000</td>
<td>-2.880</td>
<td>-2.996</td>
</tr>
<tr>
<td>Inst_13_3_2_2000</td>
<td>0</td>
<td>0.000</td>
<td>-2.171</td>
<td>-3.010</td>
</tr>
</tbody>
</table>

### Table 5. The Performance of the heuristic procedures (small-size benchmark)

<table>
<thead>
<tr>
<th>Instance</th>
<th>f</th>
<th>UB</th>
<th>H1</th>
<th>H2</th>
<th>HCl</th>
<th>HCs</th>
<th>HCc</th>
<th>VND</th>
<th>gap*</th>
<th>gap**</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inst_6_2_2_1000</td>
<td>544.78</td>
<td>637.79</td>
<td>533.07</td>
<td>533.07</td>
<td>533.071</td>
<td>544.787</td>
<td>544.787</td>
<td>544.787</td>
<td>0.00</td>
<td>0.15</td>
</tr>
<tr>
<td>Inst_7_2_2_2000</td>
<td>844.77</td>
<td>957.59</td>
<td>815.77</td>
<td>815.77</td>
<td>844.779</td>
<td>838.828</td>
<td>838.828</td>
<td>844.779</td>
<td>0.00</td>
<td>0.12</td>
</tr>
<tr>
<td>Inst_7_3_3_2000</td>
<td>1780.43</td>
<td>2186.67</td>
<td>1420.07</td>
<td>1420.07</td>
<td>1742.865</td>
<td>1460.897</td>
<td>1460.897</td>
<td>1460.999</td>
<td>0.18</td>
<td>0.33</td>
</tr>
<tr>
<td>Inst_9_2_2_2000</td>
<td>607.73</td>
<td>684.71</td>
<td>593.63</td>
<td>593.63</td>
<td>607.732</td>
<td>599.798</td>
<td>599.798</td>
<td>607.731</td>
<td>0.00</td>
<td>0.11</td>
</tr>
<tr>
<td>Inst_9_2_3_2000</td>
<td>1932.04</td>
<td>2710.12</td>
<td>1644.4</td>
<td>1473.24</td>
<td>1743.288</td>
<td>1696.424</td>
<td>1696.424</td>
<td>1932.04</td>
<td>0.00</td>
<td>0.29</td>
</tr>
<tr>
<td>Inst_10_2_3_2000</td>
<td>2310.94</td>
<td>2927.51</td>
<td>1767.09</td>
<td>1775.67</td>
<td>1775.667</td>
<td>1775.667</td>
<td>1775.667</td>
<td>2123.94</td>
<td>0.00</td>
<td>0.13</td>
</tr>
<tr>
<td>Inst_11_2_3_3000</td>
<td>2226.08</td>
<td>2346.48</td>
<td>2126.52</td>
<td>2127.934</td>
<td>2242.851</td>
<td>2162.56</td>
<td>2162.56</td>
<td>2226.08</td>
<td>0.00</td>
<td>0.05</td>
</tr>
<tr>
<td>Inst_13_2_3_1000</td>
<td>6741.72</td>
<td>3602.92</td>
<td>2194.54</td>
<td>2194.54</td>
<td>2194.54</td>
<td>2194.54</td>
<td>2194.54</td>
<td>2194.54</td>
<td>0.33</td>
<td>0.33</td>
</tr>
<tr>
<td>Inst_13_3_2_2000</td>
<td>2781.13</td>
<td>3039.55</td>
<td>2623.57</td>
<td>2623.57</td>
<td>2623.57</td>
<td>2623.57</td>
<td>2623.57</td>
<td>2623.57</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Inst_14_2_2_3000</td>
<td>546.04</td>
<td>4720.95</td>
<td>3146.87</td>
<td>3146.87</td>
<td>3146.87</td>
<td>3146.87</td>
<td>3146.87</td>
<td>3146.87</td>
<td>0.11</td>
<td>0.33</td>
</tr>
<tr>
<td>Inst_15_3_3_2000</td>
<td>4623.68</td>
<td>5609.83</td>
<td>3935.41</td>
<td>3935.41</td>
<td>3935.41</td>
<td>3935.41</td>
<td>3935.41</td>
<td>3935.41</td>
<td>0.28</td>
<td>0.28</td>
</tr>
<tr>
<td>Inst_16_3_3_2000</td>
<td>1600.13</td>
<td>2665.68</td>
<td>2665.68</td>
<td>2665.68</td>
<td>2665.68</td>
<td>2665.68</td>
<td>2665.68</td>
<td>2665.68</td>
<td>0.56</td>
<td>0.56</td>
</tr>
<tr>
<td>Inst_17_3_3_2000</td>
<td>6741.72</td>
<td>4236.92</td>
<td>4236.92</td>
<td>4236.92</td>
<td>4236.92</td>
<td>4236.92</td>
<td>4236.92</td>
<td>4236.92</td>
<td>0.37</td>
<td>0.37</td>
</tr>
<tr>
<td>Inst_18_3_3_2000</td>
<td>2686.73</td>
<td>3483.28</td>
<td>3483.28</td>
<td>3483.28</td>
<td>3483.28</td>
<td>3483.28</td>
<td>3483.28</td>
<td>3483.28</td>
<td>0.03</td>
<td>0.03</td>
</tr>
<tr>
<td>Inst_19_3_3_2000</td>
<td>2686.73</td>
<td>2296.54</td>
<td>2296.54</td>
<td>2296.54</td>
<td>2296.54</td>
<td>2296.54</td>
<td>2296.54</td>
<td>2296.54</td>
<td>0.07</td>
<td>0.07</td>
</tr>
<tr>
<td>Inst_20_3_3_2000</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>2222.072</td>
<td>0.22</td>
<td>0.22</td>
</tr>
<tr>
<td>Inst_20_3_3_2000</td>
<td>3437.68</td>
<td>2812.74</td>
<td>2812.74</td>
<td>2812.74</td>
<td>2812.74</td>
<td>2812.74</td>
<td>2812.74</td>
<td>2812.74</td>
<td>0.30</td>
<td>0.30</td>
</tr>
<tr>
<td>Instance</td>
<td>CMT_n Q_m DL L</td>
<td>f</td>
<td>UB</td>
<td>H1</td>
<td>H2</td>
<td>HcS</td>
<td>VND</td>
<td>gap*</td>
<td>gap**</td>
<td></td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------</td>
<td>----</td>
<td>-----</td>
<td>----</td>
<td>----</td>
<td>-----</td>
<td>-----</td>
<td>------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>CMT_51 Q_160 m_1 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.09</td>
<td>0.22</td>
<td></td>
</tr>
<tr>
<td>CMT_51 Q_160 m_2 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.34</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>CMT_51 Q_160 m_3 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.34</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>CMT_51 Q_160 m_4 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.34</td>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_1 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.45</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_2 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.16</td>
<td>0.17</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_3 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.32</td>
<td>0.21</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_4 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.44</td>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_5 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.49</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_6 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.41</td>
<td>0.14</td>
<td></td>
</tr>
<tr>
<td>CMT_76 Q_140 m_7 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.46</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_1 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.43</td>
<td>0.13</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_2 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.31</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_3 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.39</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_4 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.43</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_5 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.40</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>CMT_101 Q_200 m_6 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.43</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>CMT_121 Q_200 m_1 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.40</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>CMT_121 Q_200 m_2 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.11</td>
<td>0.20</td>
<td></td>
</tr>
<tr>
<td>CMT_121 Q_200 m_3 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.15</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>CMT_121 Q_200 m_4 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.42</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td>CMT_121 Q_200 m_5 DL_L2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.42</td>
<td>0.08</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_1 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.43</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_2 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.23</td>
<td>0.22</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_3 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.37</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_4 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.46</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_5 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.46</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_6 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.44</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_7 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.45</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>CMT_151 Q_200 m_8 DL_L3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.47</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_1 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.46</td>
<td>0.52</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_2 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.46</td>
<td>0.52</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_3 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.21</td>
<td>0.14</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_4 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.39</td>
<td>0.10</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_5 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.41</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_6 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.49</td>
<td>0.06</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_7 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.44</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_8 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.48</td>
<td>0.09</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_9 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.48</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td>CMT_200 Q_200 m_10 DL_L1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.48</td>
<td>0.17</td>
<td></td>
</tr>
</tbody>
</table>
7. Conclusions

In this paper, we describe a new variant of the vehicle routing problem namely the Profitable Vehicle Routing Problem with Multiple Trips. Two different strategies of sub-tours elimination constraints are used and for each strategy two different cases are defined. Thus, four mathematical models are obtained looking for optimal solutions. For large-size instances, two greedy constructive heuristics are proposed in a first solving step. Three Hill Climbing algorithms based on three neighborhood structures are developed. With a VND procedure, these methods are managed to obtain improved solutions in a second solving step of the problem. The design of these methods is based on elements of reasoning to obtain intrinsically the best possible solutions from the first iterations. Enhanced diversification was covered by a broad research approach to effectively improve the results. Experimental study shows satisfactory results for small-size instances with MILPs using some cuts. Two strategies of sub-tours elimination constraints are used representing a good idea to formulate this kind of constraints. The empirical results show the performance of the proposed constructive heuristics which provide quick solutions very close to the optimum, and also a satisfactory enhancement by using the improvement procedures. In future work, with some adjustments and the introduction of well-studied perturbation moves, the obtained results could be further refined to ensure a better optimality of solutions.

REFERENCES


[12] A. Kumar, R. Jagannathan, "Vehicle Routing with Cross Docks, Split Deliveries, and Multiple Use of Vehicles" A thesis Faculty of Auburn University, December 2011


