European Journal of Operational Research 249 (2016) 93–104

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Mathematical formulations and exact algorithm for the multitrip cumulative capacitated single-vehicle routing problem

Juan Carlos Rivera[a,b,*], H. Murat Afsar[a], Christian Prins[a]

[a] ICD-LOSI, UMR CNRS 6281, Troyes University of Technology, Troyes, France
[b] Mathematical Sciences Department, Universidad EAFIT, Medellín, Colombia

ARTICLE INFO

Article history:
Received 5 October 2014
Accepted 23 August 2015
Available online 25 September 2015

Keywords:
Multitrip cumulative capacitated
Single-vehicle routing problem
Disaster logistics
Resource constrained shortest path problem

ABSTRACT

This paper addresses the multitrip Cumulative Capacitated Single-Vehicle Routing Problem (mt-CCSVRP). In this problem inspired by disaster logistics, a single vehicle can perform successive trips to serve a set of affected sites and minimize an emergency criterion, the sum of arrival times. Two mixed integer linear programs, a flow-based model and a set partitioning model, are proposed for small instances with 20 sites. An exact algorithm for larger cases transforms the mt-CCSVRP into a resource-constrained shortest path problem where each node corresponds to one trip and the sites to visit become resources. The resulting problem can be solved via an adaptation of Bellman–Ford algorithm to a directed acyclic graph with resource constraints and a cumulative objective function. Seven dominance rules, two upper bounds and five lower bounds speed up the procedure. Computational results on instances derived from classical benchmark problems for the capacitated VRP indicate that the exact algorithm outperforms a commercial MIP solver on small instances and can solve cases with 40 sites to optimality.

© 2015 Elsevier B.V. and Association of European Operational Research Societies (EURO) within the International Federation of Operational Research Societies (IFORS). All rights reserved.

1. Introduction

A trend in the last decade is to apply operations research to facilitate logistic operations in humanitarian disasters. Several models have been proposed to cope with mitigation, preparedness, response and recovery operations, see for instance the surveys of Altay and Green III (2006) and Galindo and Batta (2013). An important issue after a disaster is to determine the transportation routes for first aids, supplies, rescue personnel or equipment from supply points to a set of destination sites, geographically scattered over the disaster region. In this context, the arrival time of relief supplies at the affected communities clearly impacts the survival rate and the suffering of victims. Hence, the total length or duration of the routes used in vehicle routing for industrial logistics must be replaced by more pertinent service-based objective functions in humanitarian logistics (Campbell, Vandenbussche, & Hermann, 2008).

The aim of this paper is to solve exactly, via a reformulation as a shortest path problem, a version of the Capacitated Vehicle Routing Problem (CVRP) inspired by the response phase of relief operations. A single vehicle with limited capacity and range is allowed to perform multiple trips and the classical objective function (total travel time or distance) becomes the sum of arrival times at affected sites. We call this problem the multitrip Cumulative Capacitated Single-Vehicle Routing Problem (mt-CCSVRP).

The paper is structured as follows. Section 2 reviews related problems. In Section 3 the problem is formally defined and two mixed integer linear models are proposed. Section 4 describes the procedure to transform the mt-CCSVRP into a resource-constrained shortest path problem in an directed acyclic graph. An ad-hoc shortest path algorithm is developed in Section 5. Computational results are presented in Section 6 for the two models and the exact method. Concluding remarks close the paper in Section 7.

2. State of the art

After a disaster, the deliveries to affected sites must take the urgency of the situation into account. Campbell et al. (2008) suggest that using service-based objective functions may better reflect the different priorities and strategic goals found in delivering humanitarian aid. The minimization of the average arrival time is more pertinent to address the emergency of humanitarian logistic operations than classical objective functions, such as the minimization of the total distance traveled. We use in this paper an equivalent objective, the sum of arrival times.

* Corresponding author at: Universidad EAFIT, Departamento de Ciencias Matemáticas, Carrera 49 N° 7 Sur-50, Medellín, Colombia.
E-mail addresses: jrivera6@eafit.edu.co (J.C. Rivera), murat.afsar@utt.fr (H. Murat Afsar), christian.prins@utt.fr (C. Prins).

http://dx.doi.org/10.1016/j.ejor.2015.08.067
0377-2217/© 2015 Elsevier B.V. and Association of European Operational Research Societies (EURO) within the International Federation of Operational Research Societies (IFORS). All rights reserved.

The sum of arrival times has been already used in some routing problems. For instance, the Minimum Latency Problem is a variation of the Traveling Salesman Problem (TSP), which consists in finding a tour starting at a depot and visiting each other node only once, in such a way that the total latency is minimized (Archer & Williamson, 2003; Blum, Chalasani, Coppersmith, Pulleyblank, Raghavan, & Sudan, 1994). The latency of a node is defined as the total distance or travel time to reach it. This problem is also known as the Delivery Man Problem (Fischetti, Laporte, & Martello, 1993) or the Traveling Repairman problem (TRP) (Jothi & Raghavachari, 2007; Tsitsiklis, 1992) because of its possible application to maintenance.

The Multiple Traveling Repairman Problem or k-TRP is a generalization of the minimum latency problem where k tours must be determined (Jothi & Raghavachari, 2007). In the Time-Dependent TSP (TDTSP), the traversal cost of an arc depends on its position in the tour (Gouveia & Voss, 1995; Lucena, 1990; Picard & Queyranne, 1978). When the objective of the TDTSP is to minimize the sum of distances traveled from the depot to each node, the problem is known as the TSP with cumulative cost or Cumulative TSP (CTSP) (Bianco, Mingozzi, & Ricciardelli, 1993).

The Cumulative Capacitated Vehicle Routing Problem (CCVRP), described by Ngueveu, Prins, and Calvo (2010), is a variant of the classical CVRP where the objective function becomes the sum of arrival times at demand nodes. The CCVRP can also be considered as the generalization of the minimum latency problem to several vehicles. These authors provide a mathematical model, several lower bounds and two memetic algorithms. Solutions are compared using classical CVRP instances from Christofides, Mingozzi, and Toth (1979), replacing the total length of the routes by the sum of arrival times, and on TRP instances from Salehipour, Sörensen, Goos, and Bräysy (2008).

The results confirm that the optimal solutions can be quite different in the CCVRP and the classical CVRP, as already observed by Campbell et al. (2008). In Euclidean versions for instance, CVRP solutions with edge crossings are suboptimal, while the elimination of these crossings often brings no improvement for the cumulative objective. More recently, Ribeiro and Laporte (2012) presented an Adaptive Large Neighborhood Search (ALNS) for the CCVRP which is compared with the memetic algorithms from Ngueveu et al. (2010), while Ke and Feng (2013) proposed a two-phase metaheuristic which improves some best known solutions from Ngueveu et al. (2010) and Ribeiro and Laporte (2012). Lysgaard and Wøhlk (2014) developed a branch and cut and price algorithm, able to solve CCVRP instances up to 69 required sites.

A comparison between the minimization of cost, maximal arrival time and average arrival time for the TSP and CVRP is given by Campbell et al. (2008). Their paper presents lower bounds, an insertion heuristic and a local search procedure. The minimization of maximal arrival time is also addressed by Applegate, Cook, Dash, and Rohe (2002) via a branch-and-cut algorithm, while Hemel, van Erk, and Jenniskens (1996) solve a practical problem aiming at minimizing the maximal tour length, using the average arrival time to break ties. Dell, Batta, and Karwan (1996) add a multi-period horizon and equity constraints.

A common assumption is that each vehicle makes a single trip, which is not realistic in disaster logistics where helicopters, for instance, can perform multiple sorties. The multitrip extension of the classical CVRP is known as the multitrip Vehicle Routing Problem (mt-VRP) and was first studied by Fleischmann (1990). Rivera, Afsar, and Prins (2015) introduce the cumulative version, called multitrip Cumulative Capacitated Vehicle Routing Problem (mt-CCVRP). They develop a flow-based model, a Multi-Start Iterated Local Search (MS-ILS) and a dominance rule for the order of routes in a multitrip. The model and the MS-ILS are evaluated on the mt-CCVRP and its particular case, the CCVRP. Rivera, Afsar, and Prins (2014) present a Multi-Start Evolutionary Local Search (MS-ELS) for the mt-CCVRP, which alternates between giant tours covering all the nodes and mt-CCVRP solutions (using a splitting procedure) and calls a variable neighborhood descent for improvement. The metaheuristic is compared with the MS-ILS of Rivera et al. (2015).

Few authors have investigated multitrip single-vehicle routing problems. Azi, Gendreau, and Potvin (2007) study a multi-trip single-vehicle routing problem with time windows (mt-SVRPTW) by enumerating all feasible trips. As it is not always possible to cover all customers, they maximize the number of visited customers and then minimize the total length. The problem is formulated as an Elementary Resource-Constrained Shortest Path Problem (ERCSPP) on an auxiliary graph and solved using a label correcting algorithm inspired by the one proposed by Feillet, Dejax, Gendreau, and Gueguen (2004).

Angel-Bello, Martínez-Salazar, and Alvarez (2013) seem to be the only authors who studied the mt-CCSVRP. They propose two mixed integer linear models for a version with a maximum number of trips. Evaluated on randomly generated instances, the best model can consistently solve instances up to 25 customers but the number of trips in the solution is limited to 2 or 3.

Our method is inspired by Azi et al. (2007) but the problems are quite different. Azi et al. consider time windows, their auxiliary graph contains circuits, and they try to visit a maximum number of sites while minimizing the total distance. In our case, there are no time windows, all sites must be visited, and the goal is to minimize the sum of arrival times at the sites. These features induce specific dominance rules and lower and upper bounds. In particular, we prove an important dominance rule which leads to an acyclic auxiliary graph.

3. Problem definition and mixed integer linear models

The mt-CCSVRP uses the sum of arrival times at sites as objective function, like in the cumulative CVRP, but involves a single vehicle which can perform more than one trip. This flexibility is necessary when the total demand exceeds the capacity of the vehicle and/or the range of the vehicle is limited. In the context of humanitarian logistics, the mt-CCSVRP models the distribution of relief supplies to a set of sites affected by a disaster, using for instance one helicopter which can do multiple sorties. After the disaster, the victims are waiting for rescue and multiple crews can be coordinated to use the vehicle full time. Nevertheless, the limited range of the vehicle imposes an upper bound on the flight time of each trip.

The problem can be defined on an complete undirected graph $G = (V, E)$. The node-set $V = \{0, \ldots, n\}$ includes one depot-node 0 and a subset $V' = V \setminus \{0\}$ of n locations affected by a disaster, called *sites* because *customers* is here inappropriate. The set E is composed of edges (i, j) with travel times w_{ij} satisfying the triangle inequality, also called *flight times*. A single vehicle of capacity Q and range L_{max} (in terms of flight time) is based at the depot. Each site $i \in V'$ has a demand q_i and a service time σ_i. It is assumed without loss of generality that $\sum_{i \in V'} q_i \geq Q$ and $q_i \leq Q, \forall\, i \in V'$.

The objective is to identify a sequence of trips $(1, 2, \ldots, v)$, called *multitrip*, such that each site is visited exactly once and the sum of arrival times is minimized (in the last trip, the return to the depot is not counted). A trip k is a cycle, starting and ending at the depot, whose total load W_k fits vehicle capacity Q and total flight time L_k does not exceed the range L_{max}. Each trip k has also a setup time (loading time) μ_k before leaving the depot, computed as the sum of service times over all sites served by the trip, weighted by a given coefficient β. The loading times and service times do not affect vehicle range but delay the arrivals at sites.

In our version, the number of trips v is a decision variable. Contrary to the mt-CVRP, the cumulative objective is sensitive to the trip ordering: Theorem 2 in Section 4 defines the optimal ordering for a given set of trips. Fig. 1 shows a small instance with $n = 5$, $Q = 20$, all $q_i = 10$, all $\sigma_i = 0$, and a three-trip feasible solution. Dashed arcs correspond to transitions between two successive trips, called

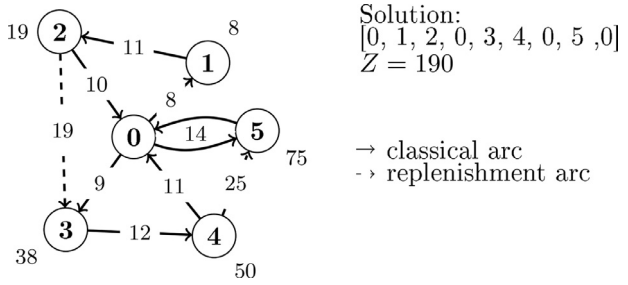

Fig. 1. Example of small mt-CCSVRP instance with one feasible solution.

replenishment arcs. The arrival time is given for each site. The cost of this solution is $Z = \sum_{i \in V'} t_i = 190$.

In the following, two mixed integer linear programs are developed. The first one is a flow-based model whose variables have no trip index. The second one is based on a set partitioning model where columns correspond to the possible routes and each site must be covered by exactly one route.

3.1. Flow-based model

The flow model derives from the one presented by Rivera, Afsar, and Prins (2013) for the mt-CCVRP, based on *replenishment arcs* (Boland, Clarke, & Nemhauser, 2000; Mak & Boland, 2000) and *arc coefficients* (Ngueveu et al., 2010). Replenishment arcs are used to replace the trips of a multitrip by a single trip: if a trip with last site i is followed by a trip with first site j, arcs $(i, 0)$ and $(0, j)$ can be replaced by one replenishment arc (i, j), see Fig. 1. The coefficient of an arc expresses the number of times its cost is counted in the objective function. Although we have here one vehicle only, the existing model must be modified to handle vehicle range, loading times and service times.

Eq. (1) defines the cost (sum of arrival times at sites) C_k for a trip $k = (0, 1, 2, \ldots, N_k, 0)$ visiting N_k sites and starting at time 0: σ_i denotes the service time at node i ($\sigma_0 = 0$) while $\mu_k = \beta \sum_{i=1}^{N_k} \sigma_i$ is the loading time. This equation can be extended to be applied to multitrips.

$$\begin{aligned} C_k &= N_k \cdot \mu_k + N_k \cdot (\sigma_0 + w_{01}) + (N_k - 1) \cdot (\sigma_1 + w_{12}) \\ &\quad + \ldots + (\sigma_{N_k - 1} + w_{N_k - 1, N_k}) \\ &= N_k \cdot \mu_k + \sum_{i=0}^{N_k - 1} (N_k - i) \cdot (\sigma_i + w_{i,i+1}) \end{aligned} \tag{1}$$

In the model, the coefficient N_k which multiplies μ_k in the first term of Eq. (1) is expressed by variables φ_{ij}, while coefficients $(N_k - i)$ are described by variables y_{ij}. Variables F_{ij} define the flow on each arc (i, j), i.e., the load of the vehicle traversing this arc. The binary variables x_{ij} are equal to 1 if and only if arc (i, j) is traversed. Due to the special nature of replenishment arcs, similar but separate variables x'_{ij} and y'_{ij} are used for these arcs. Finally, the variables t_i are used to respect vehicle range: they represent the cumulated flight time to arrive at site i since the last departure from the depot (without loading and service times), not the arrival time.

The resulting MILP is defined by Eqs. (2)–(33). The objective function (2) represents the sum of arrival times to affected sites, extending Eq. (1) to multitrips and using the variables defined above.

$$\begin{aligned} \min Z = &\sum_{i \in V} \sum_{j \in V} ((\sigma_i + w_{ij}) \cdot y_{ij} + \beta \cdot \sigma_j \cdot \varphi_{ij}) \\ &+ \sum_{i \in V'} \sum_{j \in V'} (\sigma_i + w_{i0} + w_{0j}) \cdot y'_{ij} \end{aligned} \tag{2}$$

$$\sum_{j \in V'} x_{0j} = 1 \tag{3}$$

$$\sum_{i \in V'} (x_{ij} + x'_{ij}) + x_{0j} = 1, \quad \forall\, j \in V' \tag{4}$$

$$\sum_{i \in V'} (x_{ji} + x'_{ji}) \leq 1, \quad \forall\, j \in V' \tag{5}$$

$$\sum_{j \in V} F_{ji} - \sum_{j \in V'} F_{ij} = q_i, \quad \forall\, i \in V' \tag{6}$$

$$F_{ij} \leq Q \cdot x_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{7}$$

$$F_{0j} \leq Q \cdot \left(x_{0j} + \sum_{i \in V'} x'_{ij} \right), \quad \forall\, j \in V' \tag{8}$$

$$\sum_{j \in V'} (y_{ji} + y'_{ji} - y_{ij} - y'_{ij}) + y_{0i} = 1, \quad \forall\, i \in V' \tag{9}$$

$$y_{0j} = n \cdot x_{0j}, \quad \forall\, j \in V' \tag{10}$$

$$y_{ij} \leq (n-1) \cdot x_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{11}$$

$$y'_{ij} \leq (n-1) \cdot x'_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{12}$$

$$y_{ij} \geq x_{ij}, \quad \forall\, i \in V, \quad j \in V' \tag{13}$$

$$y'_{ij} \geq x'_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{14}$$

$$y_{ij} \geq 2 \cdot x_{ij} - x_{j0}, \quad \forall\, i \in V, \quad j \in V' \tag{15}$$

$$y'_{ij} \geq 2 \cdot x'_{ij} - x_{j0}, \quad \forall\, i \in V', \quad j \in V' \tag{16}$$

$$\sum_{i \in V} \sum_{j \in V'} x_{ij} + \sum_{i \in V'} \sum_{j \in V'} x'_{ij} = n, \tag{17}$$

$$\sum_{i \in V} \sum_{j \in V'} y_{ij} + \sum_{i \in V'} \sum_{j \in V'} y'_{ij} = \frac{n \cdot (n+1)}{2}, \tag{18}$$

$$\varphi_{0j} = n \cdot x_{0j}, \quad \forall\, j \in V' \tag{19}$$

$$\varphi_{ij} \geq y'_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{20}$$

$$\varphi_{ij} \geq y_{ij}, \quad \forall\, i \in V', \quad j \in V' \tag{21}$$

$$\varphi_{ij} \geq \sum_{u \in V} \varphi_{ui} - n \cdot (1 - x_{ij}), \quad \forall\, i \in V', \quad j \in V' \tag{22}$$

$$\varphi_{ij} \leq n \cdot (x_{ij} + x'_{ij}), \quad \forall\, i \in V', \quad j \in V' \tag{23}$$

$$t_j \geq w_{0j}, \quad \forall\, j \in V' \tag{24}$$

$$t_j \geq t_i + w_{ij} - L_{max} \cdot (1 - x_{ij}), \quad \forall\, i \in V', \quad j \in V' \tag{25}$$

$$t_i + w_{i0} \leq L_{max}, \quad \forall\, i \in V' \tag{26}$$

$$x_{ij} \in \{0, 1\}, \quad \forall\, i \in V, \quad j \in V', \quad i \neq j \tag{27}$$

$$x'_{ij} \in \{0, 1\}, \quad \forall\, i \in V', \quad j \in V', \quad i \neq j \tag{28}$$

$$y_{ij} \geq 0, \quad \forall\, i \in V, \quad j \in V', \quad i \neq j \tag{29}$$

$$y'_{ij} \geq 0, \quad \forall\, i \in V', \quad j \in V', \quad i \neq j \tag{30}$$

$$F_{ij} \geq 0, \quad \forall\, i \in V, \quad j \in V', \quad i \neq j \tag{31}$$

$$\varphi_{ij} \geq 0, \quad \forall\, i \in V, \quad j \in V', \quad i \neq j \tag{32}$$

$$t_i \geq w_{0i}, \quad \forall\, i \in V' \tag{33}$$

Constraint (3) means that only one vehicle is used. Eqs. (4) and (5) indicate that exactly one arc is traversed to arrive at each site j and leave it. The last arc of the multitrip is ignored since it has no impact on the objective. Constraints (6)–(8) concern flow variables: Eq. (6) ensure that each demand is satisfied while Eqs. (7) and (8) guarantee that vehicle capacity is respected.

Constraints (9)–(12) concern arc coefficients. Constraints (9) imply that the coefficients of successive arcs decrease along a trip. Eqs. (10)–(12) limit the maximum arc coefficient. Constraints (13)–(18) are simple valid inequalities which are useful to reduce running time when the model is solved by a commercial solver. The coefficients used for loading times, the variables φ_{ij}, are constrained by Eqs. (19) –(23). The range constraint is implemented via Eqs. (24) –(26), using variables t_i.

Finally, constraints (27)–(33) define the seven groups of variables. The model has been tested on small instances and computational results are reported in Section 6.3.

3.2. Set partitioning model

The mt-CCSVRP can also be solved as a set partitioning problem, based on a directed acyclic graph (DAG) $G' = (K, H)$. K denotes the set of all feasible and non-dominated trips. It is built using a pre-computation procedure detailed in Section 4. For each trip $k \in K$, we know its cost C_k, its duration D_k, the number of visited sites N_k and binary indicators $r_k^i = 1$ if site i is visited by the trip. The arc-set H contains arc (k, k') if trip k can be followed by k' in a multitrip. Theorem 3 in Section 4 proves that the cost of a multitrip is minimized if its trips are ordered in non-decreasing order of mean duration D_k/N_k. Hence, H can be restricted to the arcs (k, k') such that $D_k/N_k < D_{k'}/N_{k'}$ and G' is acyclic. Our set partitioning formulation selects a subset of trips covering each site exactly once and defines their ordering using additional variables. The more efficient approach of Section 5 computes in G' a least-cost path visiting each site exactly once.

This model takes advantage of Eq. (34) to compute the cost of a multitrip, based on the cost, duration and number of sites of each trip. The sum of arrival times Z_p for a multitrip $p = (1, 2, \ldots, v)$ composed of v successive trips can be computed as:

$$\begin{aligned} Z_p &= C_1 + D_1 \cdot (N_2 + \cdots + N_v) + C_2 + D_2 \cdot (N_3 + \cdots + N_v) \\ &\quad + \cdots + C_{v-1} + D_{v-1} \cdot N_v + C_v \\ &= \sum_{k=1}^{v} \left(C_k + D_k \cdot \sum_{k'=k+1}^{v} N_{k'} \right) \end{aligned} \tag{34}$$

Two types of binary variables are used. The variable χ_k is equal to 1 if trip k is selected in the solution. Variables $\gamma_{kk'}$ define the relative order of the trips in the multitrip: $\gamma_{kk'} = 1$ if trip k is performed before trip k'. The set partitioning formulation for the mt-CCSVRP is given by Eqs. (35)–(41). The objective function (35) which represents the sum of arrival times is derived from Eq. (34), by adding the decision variables.

$$\min Z = \sum_{k \in K} \left(C_k \cdot \chi_k + \sum_{k' | (k,k') \in H} (D_k \cdot N_{k'} \cdot \gamma_{kk'}) \right) \tag{35}$$

$$\sum_{k \in K} r_k^i \cdot \chi_k = 1, \quad \forall\, i \in V' \tag{36}$$

$$\gamma_{kk'} \leq \chi_k, \quad \forall\, (k, k') \in H \tag{37}$$

$$\gamma_{kk'} \leq \chi_{k'}, \quad \forall\, (k, k') \in H \tag{38}$$

$$\gamma_{kk'} \geq \chi_k + \chi_{k'} - 1, \quad \forall\, (k, k') \in H \tag{39}$$

$$\chi_k \in \{0, 1\}, \quad \forall\, k \in K \tag{40}$$

$$\gamma_{kk'} \in \{0, 1\}, \quad \forall\, (k, k') \in H \tag{41}$$

Constraints (36) imply that each site is visited by exactly one trip. Constraints (37)–(39) control the order of the trips. Eq. (37) (resp. (38)) inhibits the ordering variable $\gamma_{kk'}$ when trip k (resp. k') is not used in the solution, i.e., when $\chi_k = 0$ (resp. $\chi_{k'} = 0$). Eqs. (39) impose an order between trips k and k' when both belong to the solution.

Note that arc (k, k') exists in H if and only if the mean duration of trip k is lower than the mean duration of trip k', thanks to Theorem 3 proved in the sequel. If (k, k') and (k', k'') exist in H, by transitivity (k'', k) cannot exist and we do not need to define $\gamma_{k''k}$. Therefore no cycle is possible in a solution.

Finally, Eqs. (40) and (41) define the variables. In Section 6.3 this model is tested and compared with the flow model and the exact method of Section 5 on small instances.

4. Auxiliary network for the shortest path problem

This section describes how to transform the mt-CCSVRP into a resource-constrained shortest path problem. Roughly speaking, we build a directed acyclic graph (DAG) $G' = (K, H)$, called *auxiliary network*, where each node in K (called *trip-node* to distinguish them from the nodes of the real network G) represents one trip, H contains one arc (k, k') if trip k can be followed by k' in a multitrip, and the sites become consumable resources. The pre-computation of feasible trips looks very heavy but many trips can be discarded in practice, using two dominance rules stated in Theorems 1 and 2. The process is also sensitive to the vehicle range L_{max}, as shown in the tests of Section 6.

One arc links two trip-nodes k and k' if they have no common site and if a crucial dominance property stated in Theorem 3 holds. It is this dominance rule which leads to an acyclic graph. A feasible solution is a path, called *complete path*, whose trip-nodes cover all sites.

4.1. Generation of the set of trip-nodes K

Each trip k is defined by a cost C_k, a duration D_k, a set of sites S_k, $N_k = |S_k|$, a total load W_k, a last site visited θ_k and a total flight time L_k. Recall that the latter differs from the trip duration D_k which includes service and setup times: $D_k = L_k + \sum_{i \in S_k} (1 + \beta) \cdot \sigma_i$.

K is initialized with one direct trip per site. Then, each trip k is extended by adding one site i at the end, provided i is not already visited ($i \notin S_k$) and capacity and range are satisfied ($W_k + q_i \leq Q$ and $L_k + w_{\theta_k,i} + w_{i,0} - w_{\theta_k,0} \leq L_{max}$). The number of trips with the same sites is strongly reduced using Theorem 1 and Theorem 2.

Theorem 1. *Let k and k' be two trips with sum of arrival times C_k and $C_{k'}$, durations D_k and $D_{k'}$, and visiting the same subset of sites ($S_k = S_{k'}$). Trip k dominates trip k' if $C_k \leq C_{k'}$ and $D_k \leq D_{k'}$.*

Proof. Let p' be the best complete path in G' containing k'. Its cost (sum of arrival times at sites) can be computed as (recall Eq. (34)):

$$Z_{p'} = \sum_{v=1}^{|p'|} \left(C_v + D_v \cdot \sum_{v'=v+1}^{|p'|} N_{v'} \right)$$

Now, suppose that trip k' is replaced by trip k in path p', giving path p. The difference between Z_p and $Z_{p'}$ can be computed as:

$$Z_{p'} - Z_p = C_{k'} - C_k + (D_{k'} - D_k) \cdot \sum_{v'=k'+1}^{|p'|} N_{v'}$$

Since we assumed that $C_k \leq C_{k'}$ and $D_k \leq D_{k'}$, we can conclude that $Z_{p'} \geq Z_p$, which means that trip k' is dominated by trip k. □

Theorem 2. *Let two trips k and k' with the same sites ($S_k = S_{k'}$) and the same last site ($\theta_k = \theta_{k'}$). If $C_k \leq C_{k'}$ and $D_k \leq D_{k'}$, then any trip obtained by adding a sequence v of sites at the end of trip k dominates the trip obtained by adding the same sequence at the end of k'.*

Proof. Let u and u' denote trips k and k' without the depot at the end and $\oplus$ the concatenation operator. If a sequence of sites $v = (v_1, v_2, \ldots, v_{|v|})$ is added to k and k' after the last site θ_k, the sums of arrival times for the resulting trips $(u \oplus v \oplus 0)$ and $(u' \oplus v \oplus 0)$ are:

$$Z_{u\oplus v\oplus 0} = C_k + \sum_{i=1}^{|v|}\left(D_k - w_{\theta_k,0} + w_{\theta_k,v_1} + \sum_{j=1}^{i-1}\left(\sigma_{v_{j-1}} + w_{v_{j-1},v_j}\right)\right)$$
$$= C_k + |v|\cdot(D_k - w_{\theta_k,0} + w_{\theta_k,v_1}) + C_v$$

and:

$$Z_{u'\oplus v\oplus 0} = C_{k'} + \sum_{i=1}^{|v|}\left(D_{k'} - w_{\theta_{k'},0} + w_{\theta_{k'},v_1} + \sum_{j=1}^{i-1}\left(\sigma_{v_{j-1}} + w_{v_{j-1},v_j}\right)\right)$$
$$= C_{k'} + |v|\cdot(D_{k'} - w_{\theta_{k'},0} + w_{\theta_{k'},v_1}) + C_v$$

where $C_v = \sum_{i=1}^{|v|}(\sum_{j=1}^{i-1}(\sigma_{v_{j-1}} + w_{v_{j-1},v_j}))$ is the cost of sequence v.

The difference between $Z_{u'\oplus v\oplus 0}$ and $Z_{u\oplus v\oplus 0}$ has the following expression:

$$Z_{u'\oplus v\oplus 0} - Z_{u\oplus v\oplus 0} = C_{k'} - C_k + |v|\cdot(D_{k'} - D_k)$$

Since $C_k \le C_{k'}$ and $D_k \le D_{k'}$, we can conclude that $Z_{u'\oplus v\oplus 0} \ge Z_{u\oplus v\oplus 0}$. This proves that the trip obtained by adding any sequence of sites v at the end of trip k dominates the trip built by adding the same sequence at the end of trip k. □

The two theorems concern two trips k and k' with the same sites. If $C_k \le C_{k'}$ and $D_k \le D_{k'}$ then Theorem 1 holds and, if k and k' share the same last site, Theorem 2 too. If Theorem 2 holds for $k = (1, 2, 3)$ and $k' = (2, 1, 3)$ (for instance), $k \oplus v$ will dominate $k' \oplus v$ for any sequence v of sites, e.g., (1, 2, 3, 4, 5) will dominate (2, 1, 3, 4, 5). Hence, k' can be deleted immediately. However, if k and k' satisfy Theorem 1 but have distinct last sites, e.g. (1, 2, 3) and (3, 1, 2), k dominates k' but this is not necessarily the case for extensions like (1, 2, 3, 4) and (3, 1, 2, 4). So, Theorem 1 is applied only at the end, once all possible extensions of each trip have been generated.

Note that after the precomputation of K, it is no longer necessary to verify capacity and range constraints. The same trip generation procedure is also used in the set partitioning model described in Section 3.2.

4.2. Generation of the arc-set H

H contains one arc (k, k') if trip-node k can be followed by trip-node k' in a complete path, e.g., if trip k can be followed by trip k' in a multitrip. A first condition is that k and k' visit distinct sites, i.e., $S_k \cap S_{k'} = \varnothing$. In a metaheuristic for the mt-CCVRP, Rivera et al. (2013) proved the dominance rule of Theorem 3, where the *mean duration* M_k of a trip k is defined as its total duration divided by its number of sites, i.e., $M_k = D_k/N_k$.

Theorem 3. *The cost of a multitrip is minimized by ordering its trips in non-decreasing order of mean trip duration (Rivera et al., 2013).*

According to this theorem, H can be limited to the arcs (k, k') such that $S_k \cap S_{k'} = \varnothing$ and $M_k < M_{k'}$, giving a directed acyclic graph. If $M_k = M_{k'}$ either (k, k') or (k', k) can be selected, provided no cycle is created. Moreover, by ordering the set K of trip-nodes in non-decreasing order of mean duration, we obtain a *topological sort* of G' which simplifies the shortest path algorithm. Theorem 3 can be also applied to reduce solution space in the set partitioning model of Section 3.2, by adding the constraints:

$$(M_{k'} - M_k)\cdot\gamma_{kk'} \ge 0, \quad \forall\, k \in K, \quad k' \in K \tag{42}$$

During the construction of G', some values are precomputed to ease calculations in the shortest path algorithm: the maximum number of sites per trip $N_{max} = \max\{N_k \mid k \in K\}$ and, in the original graph, the minimum cost of the arcs incident to the depot $w^0_{min} = \min\{w_{0j} \mid j \in V'\}$, the minimum service time $\sigma_{min} = \min\{\sigma_i \mid i \in V'\}$, and the minimum cost of the arcs between two sites $w_{min} = \min\{w_{ij} \mid i, j \in V'\}$.

Fig. 2 shows the auxiliary network for the instance of Fig. 1. Each square represents a trip-node k with its visited sites, under the cost C_k for a departure at time 0 and above the duration D_k. The solution (multitrip) of the first figure corresponds to the path in boldface. Its cost can be computed as $Z = (27 + 30 + 14) + 29\cdot(2 + 1) + 32\cdot 1 = 190$, following Eq. (34).

5. Shortest path algorithm

Like in Feillet et al. (2004) and Azi et al. (2007), we use an extension of Bellman–Ford algorithm with multiple labels per node to solve our resource-constrained shortest path problem. In our version, depicted in Algorithm 1, all paths are implicitly elementary since the graph is directed and acyclic: each partial path undergoes a single extension towards its feasible successors.

Algorithm 1 – Algorithm to compute a shortest path p^* in G'.

```
1:  Initial_solution (p*)
2:  for k ← 1 to |K| do
3:      Ω_k ← {k}
4:  end for
5:  for k ← 1 to |K| do
6:      for each partial path p in Ω_k do
7:          Θ_p ← Successors (K, p)
8:          for each k' ∈ Θ_p such that LB_{p⊕k'} < Z_{p*} do
9:              if (N_{p⊕k'} = n) and (C_{p⊕k'} < Z_{p*}) then
10:                 p* ← p ⊕ k'
11:             end if
12:             u ← k + 1
13:             while (u ≤ |K|) and (no path in Ω_u dominates p ⊕ k')
                do
14:                 remove from Ω_u the paths dominated by p ⊕ k'
15:                 u ← u + 1
16:             end while
17:             if u > |K| then Ω_{k'} ← Ω_{k'} ∪ {p ⊕ k'} endif
18:         end for
19:     end for
20: end for
```

The algorithm numbers the trip-nodes from 1 to $|K|$ in topological order. It defines for each trip-node $k \in K$ a set Ω_k of non-dominated

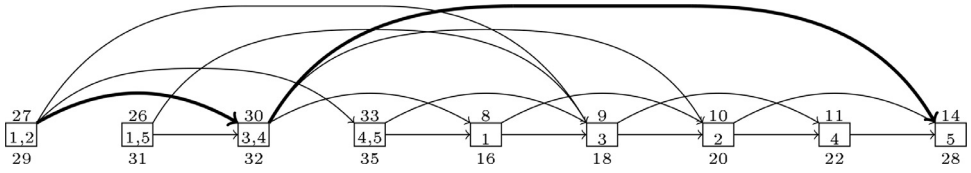

Fig. 2. Auxiliary network and example of path for the instance of Figure 1.

partial paths ending at this trip-node and updates the best solution p^* when it is improved. This solution is initialized in line 1 by the best result of two heuristics, called in the procedure *Initial_solution* described in the sequel.

Each path p is identified by a label $R_p = (C_p, D_p, S_p, N_p)$. For the sake of simplicity, we use the same notation as for the trips, but with a path index p instead of a trip index k: C_p is the total cost of the path (sum of arrival times), D_p its duration, S_p the set of sites visited and $N_p = |S_p| = \sum_{k \in p} N_k$.

In most shortest path problems on DAGs compare, feasible paths go from the first node (without predecessors) to the last (without successors). Here, other nodes may lack successors or predecessors (see Fig. 2) and a complete path may begin and end at any node. So, for each trip-node k, the algorithm initializes Ω_k with the partial path reduced to trip k, with a label $R_k = (C_k, D_k, S_k, N_k)$ (lines 2–4). Then, it inspects each trip k in topological order (line 5) and each partial path $p \in \Omega_k$ (line 6) and tries to extend it by adding one trip k' at the end. The candidates correspond to the arcs $(k, k') \in H$. The existence of (k, k') ensures by construction that trips k and k' have no common sites but some sites of k' can be visited by earlier trips on the path: this case is avoided by checking that $S_p \cap S_{k'} = \varnothing$. The set of feasible successors for path p, Θ_p, is computed in line 7 via the procedure *Successors*, which can be accelerated using the condition $N_p + N_{k'} \leq n$.

For each successor $k' \in \Theta_p$, a tentative label $R_{p \oplus k'}$ can be computed for the new path $p \oplus k'$, using the system of equations (43). Silva, Subramanian, Vidal, and Ochi (2012) defined similar equations to concatenate sequences of customers in a local search for the Cumulative TSP. We adapted them to concatenate one trip-node k' at the end of a path p.

$$\begin{aligned} C_{p \oplus k'} &= C_p + D_p \cdot N_{k'} + C_{k'} \\ D_{p \oplus k'} &= D_p + D_{k'} \\ S_{p \oplus k'} &= S_p \cup S_{k'} \\ N_{p \oplus k'} &= N_p + N_{k'} \end{aligned} \tag{43}$$

A lower bound $LB_{p \oplus k'}$ is used for the cost of the best complete path which extends $p \oplus k'$. It is computed as the maximum of five bounds presented in the next subsection. The new partial path is discarded if this bound is not smaller than the best cost Z_{p^*}. Otherwise, if $p \oplus k'$ is complete and outperforms the current best solution (line 9), p^* is updated (line 10).

Most shortest path algorithms on DAGs compare paths with the same extremities and apply dominance tests only to $\Omega_{k'}$ when one arc (k, k') is added to a partial path ending at node k. A singularity of our problem is that dominated or dominating paths may exist in sets $\Omega_{k+1}, \Omega_{k+2}, \ldots, \Omega_{|K|}$. For instance, our simplest dominance rule presented in the next subsection states that path p dominates p' if $C_p \leq C_{p'}$, $D_p \leq D_{p'}$ and $S_{p'} \subseteq S_p$. Consider a partial path p ending at trip-node k, with $S_p = \{2, 3, 4\}$, and the three next nodes such that $S_{k+1} = \{2, 3, 7\}$, $S_{k+2} = S_{k'} = \{7, 8\}$ and $S_{k+3} = \{3, 8\}$. In that case, as S_{k+1} and S_{k+3} are both included in $S_{p \oplus k'} = \{2, 3, 4, 7, 8\}$, dominated labels may exist in Ω_{k+1} and Ω_{k+3}, not only in $\Omega_{k'}$.

This is why the *while* loop lines 13–16 browses each trip-node u after k, even if the new partial path is complete, to eliminate the existing paths which are dominated. This loop stops when all trip-nodes have been inspected or as soon as an existing path dominates the new one. If the latter is not dominated (line 17), it is added to $\Omega_{k'}$.

5.1. Lower bounds

Given a partial path p, we describe now five lower bounds for the cost of the best complete path which prolongs p.

Lower Bound 1. Consider a partial path p and the best complete path $p \oplus \bar{p}$ that extends it. According to Eq. (34), the cost of the complete path is $C_{p \oplus \bar{p}} = C_p + D_p \cdot N_{\bar{p}} + C_{\bar{p}}$. As $\bar{p}$ is unknown and $C_{\bar{p}} > 0$, we can set $C_{\bar{p}}$ to 0 and, since $N_{\bar{p}} = n - N_p$, get a first lower bound $LB_p^1 = C_p + D_p \cdot (n - N_p)$ for the cost of $p \oplus \bar{p}$. This simple bound can be computed in $O(1)$.

Lower Bound 2. LB_p^1 can be improved by serving the remaining sites in a single trip k. Eq. (1) defined the cost of a trip $k = (0, 1, 2, \ldots, N_k, 0)$ as $C_k = N_k \cdot \beta \sum_{i=1}^{N_k} \sigma_i + \sum_{i=0}^{N_k - 1} (N_k - i) \cdot (\sigma_i + w_{i,i+1})$. Using the minima defined at the end of Section 4, we can set the cost of the first arc w_{01} to w_{min}^0, the cost of each arc linking two sites $w_{i,i+1}$ to w_{min}, and each service time σ_i to σ_{min}. After simplification, we get a second lower bound $LB_p^2 = LB_p^1 + (w_{min}^0 + \beta \cdot \sigma_{min} \cdot (n - N_p)) \cdot (n - N_p) + \frac{1}{2} \cdot (w_{min} + \sigma_{min}) \cdot (n - N_p) \cdot (n - N_p - 1)$. This bound dominates LB_p^1 and, as w_{min}, w_{min}^0 and σ_{min} are precomputed, it is also computable in constant time.

Lower Bound 3. The previous bound can be refined by using N_{max} to compute the minimum number of remaining trips m and a lower bound for their cost and duration. First, we have $m = \lceil (n - N_p)/N_{max} \rceil$.

Then, lower bounds for the cost and the duration of the first $m - 1$ trips, which visit at most N_{max} sites, can be written respectively $LB_c = (w_{min}^0 + \beta \cdot \sigma_{min} \cdot N_{max}) \cdot N_{max} + \frac{1}{2} \cdot (w_{min} + \sigma_{min}) \cdot N_{max} \cdot (N_{max} - 1)$ and $LB_d = 2 \cdot w_{min}^0 + w_{min} \cdot (N_{max} - 1) + (1 + \beta) \cdot \sigma_{min} \cdot N_{max}$.

As the last trip visits the $n - N_p - N_{max} \cdot (m - 1)$ remaining sites, its cost can be lower bounded as $LB_c^{last} = (w_{min}^0 + \beta \cdot \sigma_{min} \cdot (n - N_p - N_{max} \cdot (m - 1)) \cdot (n - N_p - N_{max} \cdot (m - 1)) + \frac{1}{2} \cdot (w_{min} + \sigma_{min}) \cdot (n - N_p - N_{max} \cdot (m - 1)) \cdot (n - N_p - N_{max} \cdot (m - 1) - 1)$.

Finally, the lower bound can be computed as $LB_p^3 = LB_p^1 + (m - 1) \cdot (LB_c + LB_d \cdot (n - N_p - N_{max} \cdot (m - 1))) + \frac{1}{2} \cdot LB_d \cdot N_{max} \cdot (m - 1) \cdot (m - 2) + LB_c^{last}$. It dominates LB_p^1 and LB_p^2 and can be computed in $O(1)$ too.

Lower Bound 4. This lower bound approximates the arrival times to the remaining sites. Consider a list π_p containing these sites i in non-decreasing order of distance to the depot w_{0i}. The arrival time at the ith site in π_p is approximated as $c'_i = (i \cdot \beta + i - 1) \cdot \sigma_{min} + w_{0,\pi_p(i)} \cdot \rho_{i-1} + d_i$, where $\pi_p(i)$ is the ith site in π_p, $d_i = (w_{\pi_p(i-1),0} + w_{0,\pi_p(i)}) \cdot \rho_{i-1} + d_{i-1}$, $d_0 = D_p$, $\pi_p(0) = 0$, $\rho_0 = 0$, and ρ_i a binary indicator equal to 1 if i is multiple of N_{max}. Finally, the lower bound is computed as $LB_p^4 = C_p + \sum_{i=1}^{n - N_p} c'_i$.

Note that c'_i is always smaller than the real arrival time at the ith remaining site, because a) $w_{0,\pi_p(i)} \leq \sum_{j=1}^{i} w_{\pi'_p(j-1),\pi'_p(j)}$ for any order π'_p, and b) d_i is a lower bound of the duration of trips before that site.

Lower Bound 5. In this modification of LB_p^4, the arrival time at the ith site in π_p is approximated by $c''_i = (i \cdot \beta + i - 1) \cdot \sigma_{min} + (i - \tau_i) \cdot w_{min} + d'_i$, where $d'_i = (w_{\pi_p(i-1),0} + w_{0,\pi_p(i)}) \cdot \rho_{i-1} + d'_{i-1}$, $d'_0 = D_p + w_{0,\pi_p(1)}$, and $\tau_i = \tau_{i-1} + \rho_i$ counts the estimated number of trips until the ith site ($\tau_0 = 1$). The binary indicator ρ_i is equal to 1 if, given the last position j such that $\rho_j = 1$, we have $w_{\pi_p(j),0} + (j - i) \cdot w_{min} + w_{0,\pi_p(i)} > L_{max}$ or $(i - j + 1) \mod N_{max} = 0$. Compared with LB_p^4, $w_{0,\pi_p(i)}$ in c' is replaced by $\sum_{j=1}^{i} w_{min} = i \cdot w_{min}$, which is smaller than $\sum_{j=1}^{i} w_{\pi'_p(j-1),\pi'_p(j)}$ for any order π'_p. The bound is finally computed as $LB_p^5 = C_p + \sum_{i=1}^{n - N_p} c''_i$.

The two last lower bounds can be computed in $O(n)$. The following strategy is applied in line 8 of Algorithm 1. As the bounds $LB_{p \oplus k}^1$ and $LB_{p \oplus k}^2$ are dominated by $LB_{p \oplus k}^3$, they are not used. $LB_{p \oplus k}^3$ is first computed, in $O(1)$. If $LB_{p \oplus k}^3 \geq Z_{p^*}$, the new partial path $p \oplus k$ can be discarded. Otherwise, $LB_{p \oplus k}^4$ and $LB_{p \oplus k}^5$ are computed in $O(n)$ and the best one is compared with Z_{p^*}.

5.2. Dominance rules

Section 4 describes two dominance rules to eliminate trip-nodes when building the auxiliary network. We now introduce four other

rules to discard partial paths in the shortest path algorithm. The first one (Theorem 4) extends Theorem 1 to compare two partial paths p and p' instead of two trips k and k'. It simply says that p dominates p' if it is not more costly, it does not last longer, and all sites visited by p' are also visited by p.

Theorem 4. *A partial path p dominates a partial path p' if $C_p \leq C_{p'}$, $D_p \leq D_{p'}$, and $S_{p'} \subseteq S_p$.*

Proof. Let p and p' be two partial paths with labels $R_p = (C_p, D_p, S_p, N_p)$ and $R_{p'} = (C_{p'}, D_{p'}, S_{p'}, N_{p'})$. Let $p \oplus \bar{p}$ and $p' \oplus \bar{p}'$ denote the best complete paths extending p and p', $C_{\bar{p}}$ and $C_{\bar{p}'}$ the costs of $\bar{p}$ and $\bar{p}'$, and $N_{\bar{p}} = n - N_p$ and $N_{\bar{p}'} = n - N_{p'}$ the number of sites that they visit. From Eq. (34), the costs of the two complete paths are:

$$Z_{p\oplus\bar{p}} = C_p + C_{\bar{p}} + D_p \cdot (n - N_p)$$

and

$$Z_{p'\oplus\bar{p}'} = C_{p'} + C_{\bar{p}'} + D_{p'} \cdot (n - N_{p'})$$

We prove now that $Z_{p\oplus\bar{p}} \leq Z_{p'\oplus\bar{p}'}$ by showing that each term in the first cost does not exceed the corresponding term in the second one. From the hypotheses, we know that $C_p \leq C_{p'}$ and $D_p \leq D_{p'}$. Moreover, as $S_{p'} \subseteq S_p$, we have also $N_{p'} = |S_{p'}| \leq N_p = S_p \Leftrightarrow n - N_p \leq n - N_{p'}$. Finally, as $S_{\bar{p}} \subseteq S_{\bar{p}'}$ and travel times satisfy the triangle inequality, we have $C_{\bar{p}} \leq C_{\bar{p}'}$. □

The remaining rules are deduced from Theorem 4 by computing lower bounds for the costs of the two complete paths.

Theorem 5. *A partial path p dominates a partial path p' if $S_{p'} \subseteq S_p$ and $C_p + D_p \cdot (n - N_{p'}) \leq C_{p'} + D_{p'} \cdot (n - N_{p'})$.*

Proof. Consider again two partial paths p and p' and the best complete paths $p \oplus \bar{p}$ and $p' \oplus \bar{p}'$ which extend them. According to the hypotheses:

$$C_p + D_p \cdot (n - N_{p'}) \leq C_{p'} + D_{p'} \cdot (n - N_{p'})$$

In the previous proof we saw that $n - N_p \leq n - N_{p'}$, which implies:

$$C_p + D_p \cdot (n - N_p) \leq C_{p'} + D_{p'} \cdot (n - N_{p'})$$

We also noticed that $C_{\bar{p}} \leq C_{\bar{p}'}$, so we can write:

$$C_p + D_p \cdot (n - N_p) + C_{\bar{p}} \leq C_{p'} + D_{p'} \cdot (n - N_{p'}) + C_{\bar{p}'}$$

As the two sides of this inequality correspond to $Z_{p\oplus\bar{p}}$ and $Z_{p'\oplus\bar{p}'}$ according to Eq. (34), we can conclude that path p dominates path p'. □

Theorem 6 below can be deduced from the proof of Theorem 5. It has the same statement, except that $N_{p'}$ is replaced by N_p in the left-hand side of the inequality. This theorem dominates the two previous ones.

Theorem 6. *A partial path p dominates a partial path p' if $S_{p'} \subseteq S_p$ and $C_p + D_p \cdot (n - N_p) \leq C_{p'} + D_{p'} \cdot (n - N_{p'})$.*

The last dominance rule, specified by Theorem 7, improves and generalizes the previous ones by including a lower bound on the cost to visit the sites visited by path p but not by path p', i.e., the sites in $S_p \setminus S_{p'}$.

Theorem 7. *A partial path p dominates a partial path p' if $S_{p'} \subseteq S_p$ and $C_p + D_p \cdot (n - N_p) \leq C_{p'} + D_{p'} \cdot (n - N_{p'}) + w_{min} \cdot (N_p - N_{p'}) \cdot (n - N_p) + \frac{1}{2} \cdot w_{min} \cdot (N_p - N_{p'}) \cdot (N_p - N_{p'} + 1)$.*

Proof. Like for Theorem 4, $p \oplus \bar{p}$ and $p' \oplus \bar{p}'$ are the best complete paths that extend p and p', with their costs defined by Eq. (34):

$$Z_{p\oplus\bar{p}} = C_p + C_{\bar{p}} + D_p \cdot (n - N_p)$$

and

$$Z_{p'\oplus\bar{p}'} = C_{p'} + C_{\bar{p}'} + D_{p'} \cdot (n - N_{p'})$$

As $N_{p'} \leq N_p$, path $\bar{p}'$ has to visit $N_p - N_{p'}$ more sites than $\bar{p}$ and:

$$C_{\bar{p}'} \geq C_{\bar{p}} + \frac{1}{2} \cdot w_{min} \cdot (N_p - N_{p'}) \cdot (N_p - N_{p'} + 1) + w_{min} \cdot (N_p - N_{p'}) \cdot (n - N_p)$$

where $\frac{1}{2} \cdot w_{min} \cdot (N_p - N_{p'}) \cdot (N_p - N_{p'} + 1)$ is a lower bound for the cost of visiting the $N_p - N_{p'}$ additional sites, and $w_{min} \cdot (N_p - N_{p'}) \cdot (n - N_p)$ is a lower bound for the sum of arrival time shifts at the last $(n - N_p)$ sites of $\bar{p}'$, due to the travel times to the first $N_{p'} - N_p$ sites visited by this path.

According to the hypotheses:

$$C_p + D_p \cdot (n - N_p) \leq C_{p'} + D_{p'} \cdot (n - N_{p'}) + w_{min} \cdot (N_p - N_{p'}) \cdot (n - N_p) + \frac{1}{2} \cdot w_{min} \cdot (N_p - N_{p'}) \cdot (N_p - N_{p'} + 1)$$

If we add $C_{\bar{p}}$ to both sides, we get the following inequality:

$$C_p + D_p \cdot (n - N_p) + C_{\bar{p}} \leq C_{p'} + D_{p'} \cdot (n - N_{p'}) + C_{\bar{p}} + w_{min} \cdot (N_p - N_{p'}) \cdot (n - N_p) + \frac{1}{2} \cdot w_{min} \cdot (N_p - N_{p'}) \cdot (N_p - N_{p'} + 1)$$

As shown before, the second line is not greater than $C_{\bar{p}'}$. So:

$$C_p + D_p \cdot (n - N_p) + C_{\bar{p}} \leq C_{p'} + C_{\bar{p}'} + D_{p'} \cdot (n - N_{p'})$$

which means that $Z_{p\oplus\bar{p}} \leq Z_{p'\oplus\bar{p}'}$ and p dominates p'. □

Note that when $w_{min} = 0$, Theorem 7 reduces to Theorem 6 and, when $S_p = S_{p'}$, Theorems 5, 6 and 7 give the same result. Since the last dominance rule dominates the others, it is the only one actually used in Algorithm 1.

5.3. Initial solutions

Two constructive heuristics are executed at the beginning of the algorithm. The best solution obtained is used to initialize p^*.

First heuristic. Starting with an empty path p, the first heuristic sweeps the trip-nodes in topological order (non-decreasing mean duration) and adds the incumbent trip-node k at the end of p if its sites are not already visited by the path ($S_p \cap S_k = \varnothing$). The heuristic ends when path p is complete (all sites are covered).

Second heuristic. According to the system of equations (43) and the first lower bound, the expression $C_p + C_k + D_p \cdot N_k + (D_k + D_p) \cdot (n - N_k - N_p)$ gives a lower bound for the cost of a shortest path beginning with the path $p \oplus k$. Starting again from an empty path, the second heuristic adds at each iteration the feasible trip-node k with the smallest lower bound value.

6. Computational experiments

Computational experiments have been conducted to evaluate and compare the performances of the two mathematical models and the shortest path approach. The instances used are presented in Section 6.1. Section 6.2 appraises the impact of the dominance rules of Theorem 1 and Theorem 2 on the auxiliary network generation phase. In Section 6.3, the two mathematical problems are solved using a MIP solver (CPLEX) and compared with the shortest path algorithm on instances with $n = 20$ sites. The exact approach is tested on larger instances with up to $n = 40$ in Section 6.4.

Table 1
Impact of dominance rules on auxiliary network for $n = 20$.

Instance	L_{max}	$\|K\|$	N_{max}	Strategy 1		Strategy 2	
				$\|K_g\|$	Time$_g$	$\|K_g\|$	Time$_g$
CMT$_{06}$	50	160	4	374	0.00	371	0.00
CMT$_{06}$	70	504	6	1 568	0.00	1 473	0.00
CMT$_{06}$	100	9 023	8	67, 746	6.41	43, 011	4.84
CMT$_{06}$	120	45, 599	10	736, 122	1 284.73	301, 130	285.83
CMT$_{07}$	40	173	4	424	0.00	422	0.00
CMT$_{07}$	70	596	6	1 956	0.01	1 774	0.02
CMT$_{07}$	100	9 071	8	71, 704	8.21	42, 929	5.03
CMT$_{07}$	120	38, 047	8	621, 252	1 129.94	246, 662	230.99
CMT$_{08}$	40	263	5	726	0.00	698	0.00
CMT$_{08}$	70	192	5	472	0.00	461	0.00
CMT$_{08}$	100	3 321	8	16, 778	0.32	12, 743	0.21
CMT$_{08}$	120	17, 720	9	158, 492	43.43	87, 016	20.90
CMT$_{09}$	30	209	5	524	0.00	510	0.00
CMT$_{09}$	70	636	6	2 408	0.01	2 134	0.01
CMT$_{09}$	100	10, 782	9	97, 362	13.32	54, 665	6.94
CMT$_{09}$	120	56, 866	11	1 030, 402	2 199.37	369, 435	416.75
CMT$_{10}$	30	199	6	568	0.00	529	0.00
CMT$_{10}$	70	587	6	2 192	0.01	1 933	0.01
CMT$_{10}$	100	9 220	9	72 940	7.67	43, 143	4.76
CMT$_{10}$	120	51, 047	11	774, 642	1 360.43	307, 339	309.06

6.1. Implementation and instances

The shortest path algorithm is coded in Microsoft Visual Studio C++ 2010 Professional while the mathematical models are solved using IBM ILOG CPLEX Optimization Studio version 12.5. The computer used is an Intel Core i5 PC at 2.50 GigaHertz with 4 GigaBytes of RAM and Windows 7 Professional.

The 14 classical CVRP instances from Christofides et al. (1979) (CMT instances) were taken as a basis. These benchmark problems with 50–199 customers include 7 instances with service times (instances 6 to 10, plus 13 and 14). We selected instances 6 to 10 with 50, 75, 100, 150 and 199 customers to derive 138 mt-CCSVRP instances, partitioned in four groups containing $n = 20, 25, 30$ or 40 sites, respectively. To build the set for one value of n, we take each CMT instance, assume a single vehicle, consider different values for L_{max}, keep only the depot and a subset of n sites.

The values for L_{max} are multiples of 10, between a minimum feasible value (which depends on the CMT instance) and 120. The nodes selected are the depot and the n first sites i which can be reached by the vehicle, i.e., the ones satisfying $2 \cdot w_{0i} \leq L_{max}$. The coordinates of these sites, their demands and service times, and the vehicle capacity are the same as in the original CMT file. The traveling time w_{ij} on each edge (i, j) is equal to the Euclidean distance, computed as a double-precision real number.

Finally, the loading time for each trip k in the auxiliary network is computed as $\mu_k = \beta \cdot \sum_{i \in k} \sigma_i$, where σ_i is the service time at site i and $\beta = 0.2$.

Instances 13 and 14 (100 and 120 customers) have been discarded. Contrary to the other CMT instances, their clustered nodes induce a huge number of trips. For example, instance 13 with $L_{max} = 70$ and a selection of $n = 20$ sites has already more than 20 millions feasible trips and the auxiliary network requires more than 24 hours to be generated.

6.2. Impact of dominance rules on auxiliary network construction

As explained in Section 4, Theorem 2 is used to delete dominated trips as soon as they are generated, thus avoiding useless attempts to extend them, while Theorem 1 is used at the end, once all extensions have been done. We tried two strategies to evaluate the impact of these rules. Strategy 1 involves Theorem 1 only, while Strategy 2 applies both theorems. The two strategies end by sorting the set K of non-dominated trips in non-decreasing order of mean trip duration, following Theorem 3, and building the arc-set H. Experiments were conducted on 20 instances with $n = 20$ sites and four L_{max} values: the minimum feasible multiple of 10, plus 70, 100 and 120.

Table 1 summarizes the results. The two first columns indicate the original CMT instance and the vehicle range L_{max}. Column $|K|$ reports the number of non-dominated trips obtained at the end. This number, identical for both strategies, is the number of trip-nodes in the auxiliary graph G' for the shortest path algorithm. Column N_{max} mentions the maximum number of sites per trip, to see the impact of parameter L_{max} and have an idea of the size of the solution space. The last four columns provide for each strategy the number of generated trips (columns $|K_g|$) and the total time in seconds to build the auxiliary graph (columns Time$_g$).

The table shows that Strategy 1 (Theorem 1 alone) leads to a strong reduction between the $|K_g|$ trips generated and the final $|K|$ non-dominated trips. This reduction ranges from 57% (for CMT$_{06}$) to 95% (for CMT$_{09}$), with an average of 77%. Clearly, this reduction gets stronger as L_{max} increases but, in parallel, the running time, the number of generated trips and the number of non-dominated trips grow very quickly. The maximum number of sites per trip (N_{max}) is roughly proportional to L_{max}.

The application of Theorem 2 in the second strategy brings additional savings. Compared with Strategy 1, the number of generated trips $|K_g|$ is reduced by up to 64% (for CMT$_{09}$) and by 27% on average. A decrease in running time of up to 81% (CMT$_{09}$ again) and 74% on average can also be observed: the extra time spent in applying Theorem 2 is compensated by a much smaller number of trip extensions.

Among the instances derived from CMT$_{08}$, the number of non-dominated trip-nodes $|K|$ is larger for $L_{max} = 40$ than for $L_{max} = 70$. This result is not abnormal since the n sites i extracted, which are the first to satisfy $2 \cdot w_{0i} \leq L_{max}$, are not necessarily the same for different L_{max} values.

6.3. Comparison between the mathematical models and the exact method

Table 2 compares the two mathematical models and the shortest path algorithm, reusing the 20 instances with $n = 20$ of the previous subsection. The three first columns recall the CMT instance of origin, the range L_{max} and the number of non-dominated trip-nodes $|K|$. For the exact method are reported the optimal solution value Z_{p^*} and the total running time in seconds (including network construction via Strategy 2). For each model are given the time in seconds to find an

Table 2
Results of the three solution approaches on small instances ($n = 20$).

Instance	L_{max}	$\|K\|$	Exact method		Flow-based model			Set partitioning model		
			Z_{p^*}	Time	Time	Gap_c	Gap	Time	Gap_c	Gap
CMT_{06}	50	160	5 474.61	0.12	7 200.00	5.86	0.00	2 092.72	**0.00**	0.00
CMT_{06}	70	504	5 529.37	0.12	7 200.00	8.91	0.00	7 200.00	74.44	0.00
CMT_{06}	100	9 023	5 037.30	7.76	7 200.00	8.52	0.81	–	–	–
CMT_{06}	120	45, 599	4 942.38	305.42	7 200.00	6.43	1.09	–	–	–
CMT_{07}	40	173	4 562.43	0.03	7 200.00	2.36	0.00	2 683.07	**0.00**	0.00
CMT_{07}	70	596	5 536.46	0.03	7 200.00	10.63	0.51	7 200.00	75.19	3.76
CMT_{07}	100	9 071	5 065.22	8.74	7 200.00	7.56	0.36	–	–	–
CMT_{07}	120	38, 047	5 031.30	252.98	7 200.00	6.90	0.48	–	–	–
CMT_{08}	40	263	4 616.32	0.10	7 200.00	8.00	0.00	7 200.00	27.13	1.51
CMT_{08}	70	192	7 141.29	0.10	7 200.00	10.07	0.00	2 650.01	**0.00**	0.00
CMT_{08}	100	3 321	5 272.93	1.07	7 200.00	8.53	0.56	–	–	–
CMT_{08}	120	17, 720	5 143.83	25.07	7 200.00	5.43	0.00	–	–	–
CMT_{09}	30	209	4 112.95	0.16	7 200.00	6.37	0.08	6 612.31	**0.00**	0.00
CMT_{09}	70	636	5 571.45	0.16	7 200.00	9.68	0.87	7 200.00	64.54	5.77
s CMT_{09}	100	10, 782	4 958.99	11.43	7 200.00	7.69	1.36	–	–	–
CMT_{09}	120	56, 866	4 821.66	441.59	7 200.00	3.66	0.00	–	–	–
CMT_{10}	30	199	3 953.31	0.03	7 200.00	1.18	0.00	2 341.46	**0.00**	0.00
CMT_{10}	70	587	5 627.16	0.03	7 200.00	7.73	0.00	7 200.00	65.33	6.15
CMT_{10}	100	9 220	5 035.16	8.60	7 200.00	8.75	1.45	–	–	–
CMT_{10}	120	51, 047	4 926.24	334.80	7 200.00	6.40	0.85	–	–	–

Table 3
Results of shortest path approach for $n = 25$ sites.

Instance	L_{max}	$\|K_g\|$	$\|K\|$	N_{max}	$Time_g$	Gap_0	Z_{p^*}	$Time_s$
CMT_{06}	50	1 863	657	6	0.01	3.87	7 791.63	14.20
CMT_{06}	60	932	377	5	0.00	0.12	9 000.46	3.77
CMT_{06}	70	2 170	766	6	0.01	5.93	8 521.20	2.57
CMT_{06}	80	8 220	2 502	7	0.08	2.26	7 869.37	1.71
CMT_{06}	90	29, 318	7 796	8	0.87	0.00	7 753.51	12.21
CMT_{06}	100	97, 792	22 301	8	2.49	0.50	7 678.77	74.65
CMT_{06}	110	311, 092	62 545	9	212.27	0.78	7 617.17	348.08
CMT_{06}	120	930, 583	162, 971	10	2 041.07	0.90	7 578.53	1 455.52
CMT_{07}	40	1 532	537	6	0.00	4.70	6 874.75	1.45
CMT_{07}	50	1 348	511	6	0.00	4.47	7 606.16	5.10
CMT_{07}	60	2 350	784	6	0.01	5.89	8 371.39	23.01
CMT_{07}	70	4 057	1 155	6	0.03	7.04	8 929.86	39.09
CMT_{07}	80	6 264	1 886	6	0.05	4.30	8 750.41	31.00
CMT_{07}	90	19 887	5 227	7	0.43	5.83	8 144.82	12.27
CMT_{07}	100	59 519	13 661	8	3.78	2.16	7 956.53	26.76
CMT_{07}	110	162, 762	32, 566	8	50.68	2.77	7 909.23	114.44
CMT_{07}	120	410, 258	72, 477	9	409.20	2.55	7 909.23	467.83
CMT_{08}	40	924	355	5	0.00	3.55	7 367.81	4.21
CMT_{08}	50	3 500	1 020	7	0.02	3.46	7 461.80	15.16
CMT_{08}	60	2 477	784	6	0.01	11.43	8 481.42	18.32
CMT_{08}	70	997	380	5	0.00	8.98	10 507.82	16.16
CMT_{08}	80	2 125	729	6	0.01	4.56	9 042.59	0.28
CMT_{08}	90	6 905	2 120	6	0.06	3.48	8 565.82	0.90
CMT_{08}	100	21 058	5 656	8	0.48	1.19	8 287.96	3.25
CMT_{08}	110	60 715	14 389	9	3.84	2.25	8 087.34	14.81
CMT_{08}	120	169, 345	35, 786	9	51.37	1.42	8 043.47	80.84
CMT_{09}	30	821	334	5	0.00	2.59	6 141.02	1.32
CMT_{09}	40	1 026	385	6	0.00	0.22	6 837.04	1.32
CMT_{09}	50	2 443	808	6	0.01	6.96	7 308.46	5.09
CMT_{09}	60	1 811	643	6	0.01	12.70	8 424.55	7.87
CMT_{09}	70	2 777	863	6	0.01	6.40	8 968.68	29.95
CMT_{09}	80	9 879	2 752	7	0.10	4.45	7 896.67	3.79
CMT_{09}	90	34 108	8 378	8	1.04	4.45	7 725.48	15.43
CMT_{09}	100	113, 569	24, 960	9	15.04	3.51	7 498.90	53.18
CMT_{09}	110	362, 612	73, 286	10	234.05	1.45	7 357.69	297.57
CMT_{09}	120	1 115 535	206 372	11	2 323.26	0.00	7 340.54	1 663.67
CMT_{10}	30	1 279	458	6	0.00	0.00	5 890.93	0.67
CMT_{10}	40	474	191	5	0.00	1.52	7 573.60	0.28
CMT_{10}	50	992	360	5	0.00	3.84	8 624.68	14.68
CMT_{10}	60	2 382	802	6	0.01	7.94	8 423.82	19.88
CMT_{10}	70	4 494	1 316	7	0.03	4.87	8 846.70	52.17
CMT_{10}	80	7 141	2 022	7	0.06	8.76	8 721.17	22.07
CMT_{10}	90	22 535	5 630	8	0.49	2.08	8 031.10	6.88
CMT_{10}	100	67 993	15 123	9	4.52	2.47	7 877.49	27.63
CMT_{10}	110	195 408	38 449	10	66.09	3.13	7 740.00	109.58
CMT_{10}	120	540 922	95 872	11	602.23	0.19	7 667.16	482.87

Table 4
Results of shortest path approach for $n = 30$ sites.

Instance	L_{max}	$\|K_g\|$	$\|K\|$	N_{max}	Time_g	Gap_0	Z_{P^*}	Time_s
CMT_{06}	60	2 660	930	7	0.00	10.29	11 972.47	0.22
CMT_{06}	70	4 188	1 454	6	0.02	2.99	11 884.52	0.28
CMT_{06}	80	18, 205	5 392	8	0.47	6.03	11 137.31	3.70
CMT_{06}	90	74, 422	19, 478	9	10.45	4.18	10 701.01	41.36
CMT_{06}	100	282, 931	64, 931	10	229.76	3.25	10 428.35	375.68
CMT_{06}	110	1 019 319	207, 685	10	2 951.29	2.53	10 376.84	3 232.78
CMT_{06}	120	3 356, 052	1 310, 047	11	30 959.25	2.51	10 422.68	80 226.02
CMT_{07}	50	4 875	1 531	7	0.03	3.17	10 041.72	0.32
CMT_{07}	60	5 687	1 753	7	0.05	3.86	11 357.59	0.53
CMT_{07}	70	10 454	3 006	7	0.14	5.61	11 625.83	0.97
CMT_{07}	80	40 196	9 965	8	2.11	7.06	11 293.42	10.37
CMT_{07}	90	149, 514	31, 797	8	53.32	3.57	10 832.63	80.63
CMT_{07}	100	485, 501	88, 918	9	753.47	3.61	10 738.75	595.76
CMT_{07}	110	1 37,6 717	216, 623	9	6 505.47	1.91	10 697.95	3 439.42
CMT_{07}	120	3 325, 977	476, 430	9	38 343.50	1.66	10 697.95	15 774.61
CMT_{08}	40	2 225	837	6	0.01	5.04	9 841.19	0.12
CMT_{08}	50	4 583	1 408	7	0.03	4.80	10 540.81	0.26
CMT_{08}	60	3 117	1 002	6	0.02	12.41	12 805.36	0.98
CMT_{08}	70	5 228	1 555	7	0.04	4.21	13 053.24	0.45
CMT_{08}	80	16 642	4 396	8	0.43	6.41	11 745.29	1.97
CMT_{08}	90	57, 085	13 434	9	6.25	1.55	11 262.15	15.41
CMT_{08}	100	194, 220	41, 537	9	120.87	2.43	10 936.76	138.49
CMT_{08}	110	619, 710	117, 912	10	1 219.40	3.68	10 761.69	992.23
CMT_{08}	120	1 915, 491	328, 233	11	12 512.69	2.03	10 549.61	5 973.68
CMT_{09}	40	3 335	1 077	6	0.02	3.06	9 518.48	0.21
CMT_{09}	50	3 556	1 177	6	0.02	1.67	10 309.85	0.22
CMT_{09}	60	4 561	1 466	6	0.03	11.94	11 695.71	0.33
CMT_{09}	70	4 639	1 503	6	0.03	8.87	12 523.25	0.38
CMT_{09}	80	19 261	5 512	7	0.45	7.51	11 362.75	4.09
CMT_{09}	90	77 839	19 836	9	8.38	5.58	10 709.45	39.11
CMT_{09}	100	298, 288	67, 979	10	183.63	1.88	10 546.39	386.07
CMT_{09}	110	1 084, 926	226, 277	11	2 506.97	3.36	10 396.25	3 511.24
CMT_{09}	120	3 812, 339	732, 867	12	33 255.76	3.57	10 239.78	26 629.38
CMT_{10}	40	1 328	459	6	0.00	2.29	9 863.88	0.05
CMT_{10}	50	1 753	634	6	0.00	4.05	11 715.45	0.65
CMT_{10}	60	6 038	1 855	7	0.05	3.22	10 914.47	0.43
CMT_{10}	70	15, 136	4 105	7	0.32	8.39	11 260.09	1.94
CMT_{10}	80	40 665	10, 197	9	2.15	3.07	11 206.42	8.09
CMT_{10}	90	160, 153	35, 250	10	56.59	5.85	10 662.64	87.47
CMT_{10}	100	583, 174	112, 859	11	930.87	3.51	10 463.74	734.42
CMT_{10}	110	1 981, 764	336, 220	12	10 573.35	2.14	10 301.81	5 338.83
CMT_{10}	120	6 307 617	944 922	12	105 856.07	2.11	10 269.46	36 075.79

optimal solution (limited to two hours), the percentage gap between the best integer solution found and the best lower bound reported by CPLEX (Gap_c) and the percentage gap between the best integer solution found and the optimal solution. Null gaps are highlighted in boldface while dashes indicate memory overflows in CPLEX.

The results show that the shortest path approach is much more efficient than a direct resolution of the two mixed integer linear programs: it solves all instances to optimality in 69 seconds on average and in 441 seconds (7.4 minutes) in the worst case. Nevertheless, the number of trip-nodes in the auxiliary network and the total running time grow quickly with L_{max}. A comparison with the previous table reveals than the fraction of time devoted to the auxiliary network construction increases in parallel, reaching 92.3% for the last instance (309.06 seconds out of 334.80).

Although the instances considered are small (20 sites), the flow-based model retrieves only nine of the optima found by the shortest path algorithm and CPLEX has no proof of optimality since its branch-and-cut is always interrupted after two hours. The CPLEX gaps (column Gap_c) varies between 1% and 10% but is not sensitive to the parameter L_{max}.

CPLEX behaves differently on the set partitioning formulation, due to its large number of colums: $|K|$ binary variables are used to select the trips, plus $O(|K|^2)$ to enforce the dominance relation for the order of these trips in the multitrip. 5 instances with at most 209 trip-nodes are solved to optimality. The time limit is reached and poor integer solutions are returned for 5 other instances with up to 636 trip-nodes. The 10 remaining instances from 3 321 trip-nodes onwards raise an "out of memory" error.

Contrary to the other instances, on the data files derived from CMT_{08}, the set partitioning formulation obtains an optimal solution when $L_{max} = 70$ but not for $L_{max} = 40$. These results can be explained by the number of non-dominated trip-nodes, which is larger in the instance with $L_{max} = 40$.

6.4. Test of the exact method on larger instances

The previous subsections have shown that the shortest path approach is fast enough for $n = 20$ (one minute on average) to envisage solving larger instances. Tables 3, 4 and 5 respectively gather our results for 25, 30 and 40 sites. Most column headers have been already defined. The number of trips generated $|K_g|$ and the final number of non-dominated trips $|K|$ come again from Strategy 2. We distinguish now between the time to generate the auxiliary network (Time_g) and the time to determine the shortest path (Time_s), and provide the percentage gap between the initial solution and the optimum (Gap_0). Concerning vehicle range L_{max}, all multiples of 10 between the minimum feasible value and 120 are considered for $n \in \{25, 30\}$. For $n = 40$, L_{max} has been limited to 100 otherwise the network is too large.

The results in Tables 3, 4 and 5 confirm the impact of the dominance rule of Theorem 1. If $|K_g|$ and $|K|$ are compared, the number of trip-nodes in the auxiliary network is reduced by 59–82% for

Table 5
Results of shortest path approach for $n = 40$ sites.

Instance	L_{max}	$\|K_g\|$	$\|K\|$	N_{max}	$Time_g$	Gap_0	$Z_{P'}$	$Time_s$
CMT_{06}	70	14, 274	4 397	7	0.28	3.50	20 867.69	2.94
CMT_{06}	80	67, 730	18, 032	9	7.91	5.84	19 361.03	49.16
CMT_{06}	90	318, 535	76, 182	10	288.26	3.50	18 835.61	720.29
CMT_{06}	100	1 435 ,280	306, 674	11	7 568.10	4.15	18 203.66	9 493.35
CMT_{07}	50	29, 750	7 435	7	1.65	8.95	17 516.44	6.51
CMT_{07}	60	37, 199	9 568	7	1.88	5.04	18 703.93	14.47
CMT_{07}	70	44, 396	11, 204	8	2.39	8.00	19 762.98	27.88
CMT_{07}	80	181, 916	41, 388	8	83.44	4.80	18 866.02	246.15
CMT_{07}	90	755, 087	149, 248	9	1 954.73	2.49	18 398.30	3 258.86
CMT_{07}	100	2 719, 984	475, 916	9	26 683.87	2.64	18 223.89	39 534.99
CMT_{08}	50	7 176	2 385	7	0.06	3.12	18 399.02	1.04
CMT_{08}	60	22, 509	5 820	8	0.63	8.35	20 897.66	7.25
CMT_{08}	70	11, 287	3 199	7	0.18	5.46	22 532.44	2.31
CMT_{08}	80	52, 230	13, 564	8	3.80	2.22	20 443.41	22.80
CMT_{08}	90	181, 591	41, 935	9	91.08	4.73	19 913.11	199.83
CMT_{08}	100	722, 857	152, 764	10	1 927.96	5.27	19 231.59	2 662.42
CMT_{09}	40	12, 293	3 705	7	0.22	7.18	16 130.70	26.82
CMT_{09}	50	37, 699	9 437	9	2.38	4.16	16 569.72	13.60
CMT_{09}	60	28, 283	7 914	8	0.91	5.47	18 280.33	9.36
CMT_{09}	70	22, 859	6 329	7	0.73	5.77	21 818.16	5.15
CMT_{09}	80	98, 199	24, 222	9	20.65	7.32	19 149.11	69.57
CMT_{09}	90	441, 127	97, 765	10	621.76	6.82	18 574.58	903.67
CMT_{09}	100	1 926, 620	386, 437	11	15 106.90	6.60	18 142.65	14 513.45
CMT_{10}	40	6 882	2 058	7	0.07	3.14	16 073.16	1.35
CMT_{10}	50	13, 216	3 763	7	0.30	2.89	18 061.52	33.07
CMT_{10}	60	17, 778	5 282	8	0.34	6.27	18 369.49	4.36
CMT_{10}	70	35, 263	9 490	8	1.52	3.34	19 551.22	14.08
CMT_{10}	80	171, 076	40, 270	10	54.32	6.18	18 595.38	185.93
CMT_{10}	90	835, 757	177, 535	11	1 870.86	3.77	18 120.34	3 146.68
CMT_{10}	100	3 870, 005	746, 480	11	34 203.30	2.32	17 719.54	45 118.58

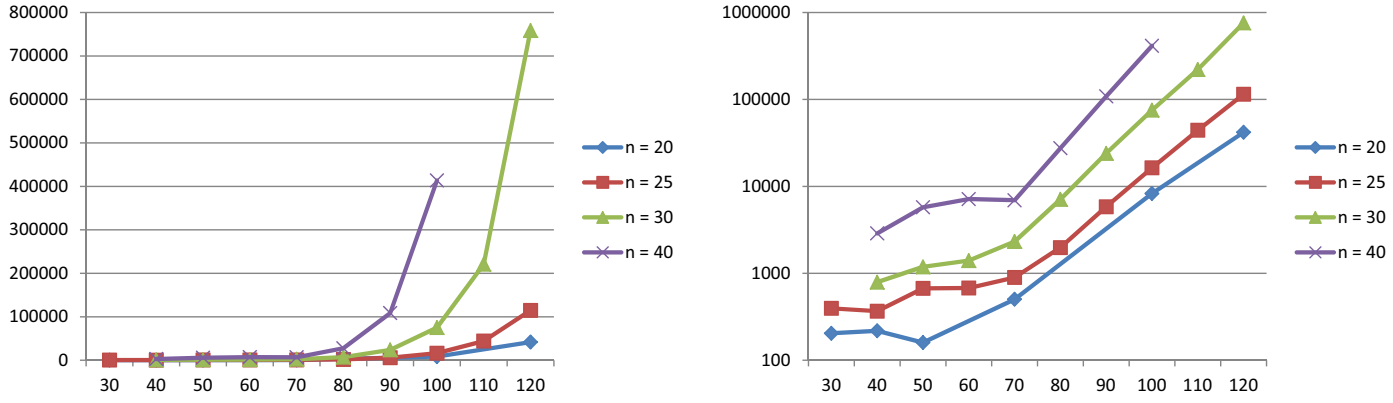

Fig. 3. Impact of n and L_{max} on $|K|$ (linear and logarithmic scales).

instances with 25 sites, 60–85% for $n = 30$, and 66–82% for $n = 40$. These percentage are not significantly affected by the number of sites but tend to increase with L_{max}. For most instances, a greater value of $|K_g|$ or $|K|$ implies a greater value of the maximum number of sites per trip N_{max}, but the latter can stagnate because of capacity constraints, see for example instance CMT_{07} in Table 4.

The two heuristics used at the beginning yield very good solutions on average, since Gap_0 varies between 0% and 12.7%, with an average value of 4.31%. Nevertheless, a better initial solution does not always imply a faster resolution. Consider in Table 5 instance CMT_{07} with $L_{max} = 90$ and CMT_{08} with $L_{max} = 100$. They have comparable networks (149, 248 and 152, 764 non-dominated trips). However, the first one which has a better initial gap (2.49% vs. 5.27%) takes longer to be solved (3 258 seconds vs. 2 662 seconds).

The time to generate the auxiliary network and compute the shortest path increases quickly with n and L_{max}. For instances with the largest L_{max} values, constructing the network can be more time-consuming than determinating the min-cost path, like for CMT_{10} with $L_{max} = 120$ in Table 4. Figs. 3 and 4 illustrate respectively the number of non-dominated trip-nodes ($|K|$) and the total running time ($Time_g$ + $Time_s$) for the different values considered for n and L_{max}. Each point is averaged over all instances with the same values of n and L_{max}. The left part of each figure uses a linear scale for the Y-axis while the right part is logarithmic. In the logarithmic plots, the curves are almost linear when L_{max} is large enough, indicating an exponential growth.

7. Conclusions and future work

This paper has presented a new version of the mt-CCVRP, in which only one vehicle is available and the range of the vehicle represents a very restrictive constraint. This combinatorial optimization problem constitutes a good way to model the delivery of relief supplies after a disaster, since the objective function takes into account the urgency of the situation.

Two mathematical models have been compared with an exact approach based on a reformulation of the problem as a resource-constrained shortest path problem. While a commercial MIP software solves only a minority of instances for 20 sites, the exact method

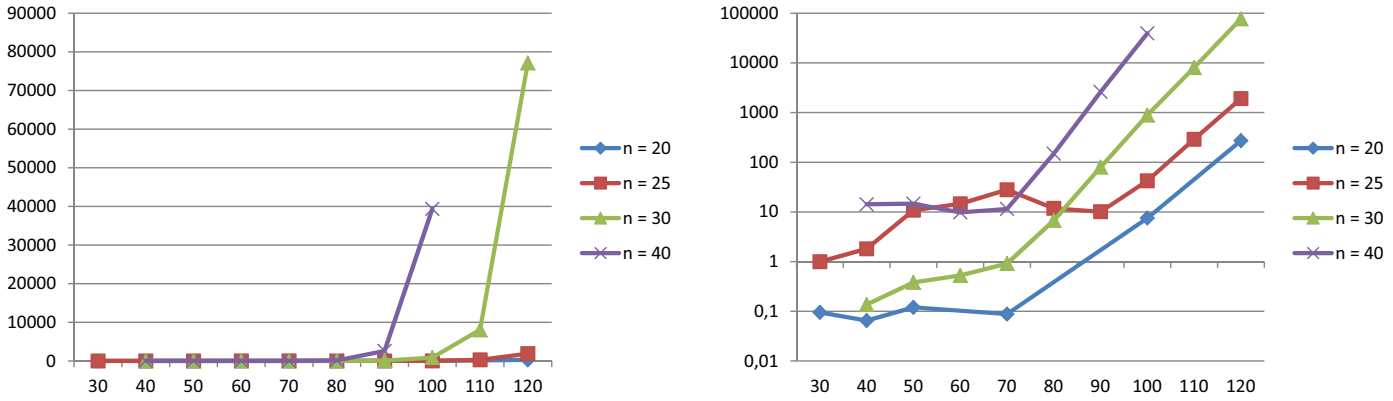

Fig. 4. Impact of n and L_{max} on total running time (linear and logarithmic scales).

can tackle instances with up to 40 sites, thanks to good initial solutions, dominance rules and lower bounds which accelerate the solution procedure. Like many dynamic programming approaches, our method reaches its limits when the state-graph is too big, which happens in our case for large values of the vehicle range (L_{max}).

Among possible future research directions, the exact approach could be used in a column generation scheme for the mt-CCVRP. Cumulative versions of other node-routing problems, such as the Generalized VRP or the Selective VRP, are worth studying to model different distribution policies in humanitarian logistics. Similar objectives like the sum of weighted arrival times are also interesting to reflect site priorities.

Acknowledgment

This research is funded in part by a doctoral grant from the French Ministry of Research and Higher Education (J.C. Rivera).

References

Altay, N., & Green III, W. G. (2006). OR/MS research in disaster operations management. *European Journal of Operational Research, 175*(1), 475–493.

Angel-Bello, F., Martínez-Salazar, I., & Alvarez, A. (2013). Minimizing waiting times in a route design problem with multiple use of a single vehicle. *Electronic Notes in Discrete Mathematics, 41*, 269–276.

Applegate, D., Cook, W., Dash, S., & Rohe, A. (2002). Solution of a min–max vehicle routing problem. *INFORMS Journal on Computing, 14*, 132–143.

Archer, A., & Williamson, D. P. (2003). Faster approximation algorithms for the minimum latency problem. In *Proceedings of the fourteenth annual acm-siam symposium on discrete algorithms (soda)*.

Azi, N., Gendreau, M., & Potvin, J.-Y. (2007). An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. *European Journal of Operational Research, 178*(3), 755–766. http://dx.doi.org/10.1016/j.ejor.2006.02.019.

Bianco, L., Mingozzi, A., & Ricciardelli, S. (1993). The traveling salesman problem with cumulative costs. *Networks, 23*(2), 81–91.

Blum, A., Chalasani, P., Coppersmith, D., Pulleyblank, B., Raghavan, P., & Sudan, M. (1994). The minimum latency problem. In *Proceedings of the 26th annual acm symposium on theory of computing (stoc'94)* (pp. 163–171).

Boland, N., Clarke, L., & Nemhauser, G. (2000). The asymmetric traveling salesman problem with replenishment arcs. *European Journal of Operational Research, 123*(2), 408–427.

Campbell, A. M., Vandenbussche, D., & Hermann, W. (2008). Routing for relief efforts. *Transportation Science, 42*(2), 127–145.

Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, & L. Sandi (Eds.), *Proceedings of the Combinatorial optimization* (pp. 315–338). Wiley, Chichester.

Dell, R. F., Batta, R., & Karwan, M. H. (1996). The multiple vehicle TSP with time windows and equity constraints over a multiple day horizon. *Transportation Science, 30*(2), 120–133.

Feillet, D., Dejax, P., Gendreau, M., & Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: application to some vehicle routing problems. *Networks, 44*(3), 216–229. doi:10.1002/net.20033.

Fischetti, M., Laporte, G., & Martello, S. (1993). The delivery man problem and cumulative matroids. *Operations Research, 41*(6), 1055–1064.

Fleischmann, B. (1990). The vehicle routing problem with multiple use of vehicles. *Working paper*. Department of Economics, University of Hamburg.

Galindo, G., & Batta, R. (2013). Review of recent developments in OR/MS research in disaster operations management. *European Journal of Operational Research, 230*(2), 201–211.

Gouveia, L., & Voss, S. (1995). A classification of formulations for the (time-dependent) traveling salesman problem. *European Journal of Operational Research, 83*(1), 69–82.

Hemel, T., van Erk, S. Jenniskens, P.(1996). The Manhattan project, URL http://www.win.tue.nl/whizzkids/1996/tsp.html.

Jothi, R., & Raghavachari, B. (2007). Approximating the k-traveling repairman problem with repairtimes. *Journal of Discrete Algorithms, 5*(2), 293–303.

Ke, L., & Feng, Z. (2013). A two-phase metaheuristic for the cumulative capacitated vehicle routing problem. *Computers & OR, 40*, 633–638.

Lucena, A. (1990). Time-dependent traveling salesman problem: The deliveryman case. *Networks, 20*, 753–763.

Lysgaard, J., & Wøhlk, S. (2014). A branch-and-cut-and-price algorithm for the cumulative capacitated vehicle routing problem. *European Journal of Operational Research, 236*(3), 800–810.

Mak, V., & Boland, N. (2000). Heuristic approaches to the asymmetric travelling salesman problem with replenishment arcs. *International Transactions in Operational Research, 7*(4-5), 431–447.

Ngueveu, S. U., Prins, C., & Calvo, R. W. (2010). An effective memetic algorithm for the cumulative capacitated vehicle routing problem. *Computers & Operations Research, 37*(11), 1877–1885.

Picard, J.-C., & Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research, 26*(1), 86–110.

Ribeiro, G. M., & Laporte, G. (2012). An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research, 39*(3), 728–735.

Rivera, J., Afsar, H., & Prins, C. (2014). Multistart evolutionary local search for a disaster relief problem. In P. Legrand, M.-M. Corsini, J.-K. Hao, N. Monmarché, E. Lutton, & M. Schoenauer (Eds.), *Artificial evolution*. In *Lecture Notes in Computer Science: 8752* (pp. 129–141). Springer International Publishing.

Rivera, J., Afsar, H., & Prins, C. (2015). A multistart iterated local search for the multitrip cumulative capacitated vehicle routing problem. *Computational Optimization and Applications, 61*(1), 159–187.

Rivera, J. C., Afsar, M. H., & Prins, C. (2013). A multi-start iterated local search for the multitrip cumulative capacitated vehicle routing problem. *Technical Report UTT-LOSI-13001*. Troyes University of Technology.

Salehipour, A., Sörensen, K., Goos, P., & Bräysy, O. (2008). An efficient GRASP + VND metaheuristic for the traveling repairman problem. *Working paper*. University of Antwerp, Faculty of Applied Economics.

Silva, M. M., Subramanian, A., Vidal, T., & Ochi, L. S. (2012). A simple and effective metaheuristic for the minimum latency problem. *European Journal of Operational Research, 221*(3), 513–520.

Tsitsiklis, J. N. (1992). Special cases of traveling salesman and repairman problems with time windows. *Networks, 22*, 263–282.