CIRRELT-2022-11

Iterative Time-Decomposition Matheuristic for the Biomedical Sample Transportation Problem

Chiara Mazzantia
Ana María Anaya-Arenas
Valérie Bélanger
Ettore Lanzarone
Angel Ruiz

April 2022

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2022-003

Bureau de Montréal
Université de Montréal
C.P. 6 128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

Iterative Time-Decomposition Matheuristic for the Biomedical Sample Transportation Problem

Chiara Mazzanti[1,*], Ana María Anaya-Arenas[2,3], Valérie Bélanger[2,4], Ettore Lanzarone[1,5], Angel Ruiz[2,6]

1. Institute for Applied Mathematics and Information Technology (IMATI), Consiglio Nazionale delle Ricerche (CNR), Milan, Italy
2. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)
3. Département d'analytique, opérations et technologies de l'information, École des sciences de la gestion, Université du Québec à Montréal, Québec, Montréal
4. Département de gestion des opérations et de la logistique, HEC, Montréal, Canada
5. Department of Management, Information and Production Engineering, University of Bergamo, Dalmine (Bg), Italy
6. Département d'opérations et systèmes de décision, Faculté des sciences de l'administration, Université Laval, Québec, Canada

Abstract. This paper proposes an iterative time-decomposition matheuristic for solving the biomedical sample transportation problem (BSTP), which is a routing problem with multiple and interdependent visits in the context of healthcare services. In this problem, each healthcare or specimen collection center collects biomedical samples from individuals. Because the lifespan of a specimen lasts only a few hours from collection to analysis, several collection centers must be visited more than once a day to collect the specimens and ensure that they are analyzed before perishing. Setting a maximum time to analyze the samples imposes a time interdependency between visits to the same center and the maximum duration of their corresponding routes. This is a complex routing problem, and commercial solvers have been inefficient at solving it. Hence, we propose an algorithm that uses a time-decomposition technique to reduce the interdependency and apply a Fix-&-Optimize technique to solve the problem efficiently. The matheuristic proves to be efficient in solving a set of real-life instances with high interdependency requirements from the Quebec laboratory network under the management of the Ministère de la Santé et des Services sociaux (Ministry of Health and Social Services).

Keywords: vehicle routing problem, biomedical samples transportation problem, healthcare logistics, interdependency, synchronization, blood transportation, Rolling Horizon Heuristic, Fix-&-Optimize

Acknowledgments: The authors thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for its financial support under Grants no RGPIN-2018-04609 and the Fonds de Recherche du Québec Nature et Technologies (FRQNT) under grant 2021-NC-282544.

---

* Corresponding author: Anaya-Arenas.Ana_Maria@uqam.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2022

1 Introduction

Healthcare providers must usually deploy large facility networks to ensure proximity to patients. However, the increase in facilities leads to several challenges from a managerial perspective, especially when involving high-tech and expensive equipment, because demand consolidation and return to scale opportunities are reduced. Therefore, healthcare managers should be careful when designing services to balance the requirements of proximity and system costs.

The management of biomedical samples facing these challenges requires the planning of logistic activities from the collection of a specimen to its analysis. Managers intend to multiply the collection points over the territory to facilitate service access for individuals. Moreover, they intend to minimize the cost of deployable laboratory equipment and maximize its usage. Both goals are achievable by planning adequate networks that collect and transport samples from the various service points to a reduced number of laboratories in which they are analyzed. However, having a decentralized collection and centralized analysis requires complex logistics that must maintain costs as low as possible and respect a set of practical restrictions, the short lifetime of biomedical samples, being one, if not the most difficult restriction, among them.

The Biomedical Sample Transportation Problem (BSTP) arising in the Province of Quebec, Canada inspired our study. It was first introduced in Anaya-Arenas et al. (2016) as a vehicle routing problem aimed at planning multiple and independent visits at collection centers (SCCs) to ensure that samples are transported to designated regional analysis centers (or Labs) within strict time limits given by the short lifespan of the samples. Our study considers a more realistic version that differs from that described in Anaya-Arenas et al. (2016) in two essential points. First, we assume that each SCC is flexible in deciding its opening time, although the total collection time is fixed and known a priori. Second, unlike in Anaya-Arenas et al. (2016), which hypothesizes that samples begin perishing once they leave the collection facility, we acknowledge that a sample begins deteriorating as soon as it is collected, resetting the latest time at which it can be analyzed at a Lab. As in Anaya-Arenas et al. (2016), samples are accumulated until a transport gathers and brings them to the lab. Because several SCC are intended to be visited sequentially in a route, the short lifespan of the samples implies that 1) a single SCC requires more than one visit on the same day and 2) the time that samples can be accumulated at the SCC waiting to be transported, and the time required by a vehicle to transport them to a Lab are limited and linked.

This study focuses on the interlaced routing and schedule decisions of BSTP and proposes an iterative time-decomposition matheuristic for efficiently solving real instances of the problem. The classical *rolling horizon* (RH) strategy inspired the matheuristic to separate the problem into a collection of sets of smaller, easier to solve subproblems at each iteration. The subproblems are solved using the mathematical formulation of BSTP combined with effective *Fix-&-Optimize* (F&O) strategies that speed up the search while guaranteeing an efficient exploration of the solution space associated with each subproblem.

The remainder of this paper is organized as follows. Section 2 summarizes relevant studies and discusses this research with respect to their contributions. Section 3 identifies the modelling differences between the biomedical context studied here and previous works dealing with the specimens and blood transports, and then adapts the mathematical formulation of BTSP to address such differences. Section 4 describes the matheuristic, and Section 5 reports the numerical results. Finally, Section 6 summarizes the main contributions of this study and suggests future research directions.

2 Literature Review

The transportation of blood and other specimens is a classical logistic problem related to the delivery of health services at the local, regional, and national levels (Brailsford & Vissers, 2011). Strong time restrictions and/or precedence constraints characterize these problems, which are at least partially also present in other healthcare logistics cases. The next paragraphs review the first contributions in home-care planning problems that address synchronization and/or time interdependencies on visits to patients. The second part of this section reviews the contributions to specimen and blood transportation.

Interdependency in the visit schedule is appropriate for many service applications in which some type of synchronization (or temporal precedence) constraint must be imposed. The problem has been study under the name *vehicle routing problem with time windows and synchronization (VRPTW-Syn)* (refer to Bredström & Rönnqvist, 2008; Dohn et al., 2011; Drexl, 2012; Afifi et al., 2016, for more details). This problem has been applied in the context of forestry, the technician routing problem, and home healthcare, among others (e.g., Bredström & Rönnqvist, 2008; Rousseau et al., 2013; Euchi et al., 2021; Melachrinoudis et al., 2007). In this problem, synchronization restrictions have often been studied to either limit or exceed the time between two visits to the same node (e.g., a patient must be simultaneously visited by two independent specialists or requiring that the cleaning service (first visit) be scheduled one hour before the medical service (second visit)). Owing to the complexity of the problem, past contributions to VRPTW-Syn have primarily proposed heuristic algorithms whose performance have often been tested over the test set of Bredström & Rönnqvist (2008). This test bed contains up to 80 visit instances with fixed and independent time windows, and in 10% of them a type of synchronization is requested.

An important healthcare application for VRPTW-Syn is the routing of resources for home healthcare (HHC). For example, Ait Haddadene et al. (2016) proposed a GRASPxILS metaheuristic that includes budgetary restrictions. Kergosien et al. (2014) analyzed the routing of nurses considering drop-off of samples. Liu et al. (2013) proposed the pickup and delivery of goods, but the lifespan was longer than a day (single visit per customers without a time limit). Decerle et al. (2018) proposed a memetic algorithm for solving an HHC problem in France, and Frifita & Masmoudi (2020) proposed metaheuristics to solve the problem, including several specialties in the scheduling . Melachrinoudis et al. (2007), in particular, set a dial-a-ride problem for a healthcare organization in Boston (USA), and proposed a tabu search algorithm to solve real-life instances of up to 5 (independent) transportation requests. BSTP shares the synchronization challenge of said problems, but in the case of BSTP, a visit to one SCC impacts its next visit well as the whole collection route in which it will be done by imposing the latest arrival time to the lab. Moreover, our problem does not consider any time windows for the visits, which increases the complexity and interdependency in the decision-making.

Furthermore, a series of logistic problems with strict temporal restrictions have been studied in the blood supply chain (BSC) literature because of the perishability of products (see Osorio et al., 2015; Pirabán et al., 2019; Baş et al., 2016 for comprehensive reviews). Since the first studies in the 1960's, over 200 papers have been published in the field (Pirabán et al., 2019). Recent studies have analyzed the design of the entire BSC network, considering the multi-echelon aspect of the problem, as it includes the collection, production, inventory, and distribution of blood products with a lifespan of a few days (e.g., Yousefi Nejad Attari et al., 2019; Ghandforoush & Sen, 2010; Baş et al., 2018; Araújo et al., 2020). However, such contributions rarely include transportation planning, or (if planned) it uses shuttles that execute several trips to a single collection point without routing.

Considering the collection stage of the BSC, some studies aim to plan collection routes to maximize the number of processed samples and minimize transportation costs (e.g., Yücel et al., 2013), whereas others, such as Mobasher et al. (2015) coordinate the appointment schedule with the transportation planning to maximize platelet production. Şahinyazan et al. (2015) proposed a two-stage IP based heuristic to simultaneously determine the sequence and the length of stops at mobile clinics over a week, while shuttles bring samples to the lab at the end of each day. Zahiri et al. (2018) included the freshness of delivered products in their objective (maximizing the active shelf life after delivery) and specifically tracked the moment at which samples were collected. Nonetheless, all these studies assume the lifespan of produces are one day or longer, removing the need for more than one visit per day and, therefore, the dependency between routes.

Anaya-Arenas et al. (2016) introduced BSTP inspired by the needs of Ministry of Health and Social Services of Quebec to transport samples from the SCCs to the labs. The short lifespan of the samples was addressed using independent hard time windows and a limited route duration. Naji-Azimi et al. (2016) planned the de-synchronization of trucks arrivals to the lab in the same context, and Zabinsky et al. (2020) presented the same problem but applied a case study from Washington Medical Center. In these three studies, temporal constraints were present, but only one visit was required per customer. Therefore, no precedence or synchronization is necessary between visits.

Doerner et al. (2008) were one of the few to explicitly present the interdependency created by the deterioration of the samples, including the time restrictions and the precedence constraints. In their study, a savings and greedy construction heuristic was proposed to solve instances involving up to 15 customers. Elalouf et al. (2018), studied a similar problem and solved cases of up to 11 customers. Assumingly, when interdependency is considered, the current state-of-the-art heuristics focus on solving instances of less than 20 customers with more than a single visit. Finally, Anaya-Arenas et al. (2021) proposed an iterative local search algorithm to solve BSTP. In their study, the lifespan of the samples was divided by the maximum waiting time at each SCC (Δ_{max}) and maximum routing time (T_{max}). This study proposes a more realistic accountability of the lifespan from the moment a sample is drawn until the sample arrives at the Lab. The ILS algorithm of Anaya-Arenas et al. (2021) uses the Δ_{max} parameter to impose fictive time windows and then uses concatenation techniques to solve the problem. However, this is inapplicable to the generalization of the problem presented. Therefore, this study proposes an iterative time-decomposition matheuristic addressing the challenges of timing and routing interdependency in a new way.

3 Problem Modelling and Formulation

This section describes the characteristics of BTSP and proposes its mathematical formulation. Section 3.1 formalizes the problem, emphasizing how the short lifespans of samples require the planning of several visits to the same SCC and the resulting intertwined links between them.

Subsequently, Section 3.2 presents a mathematical formulation of the problem.

3.1 Problem statement

The addressed BSTP is defined over an administrative region that contains a set $C = \{c_1, c_2, \ldots, c_n\}$ with n SCCs in which samples are drawn from individuals and accumulated. At defined moments, the SCC is visited by a vehicle that collects the drawn samples and transports them to the corresponding laboratory, referred to as the Lab, where the samples will be analyzed. Different parameters characterize every SCC c_g ($g \in [1, \ldots, n]$). First, the collection period of an SCC c_g is described by its length O_g and time window for the opening time a_g, defined as $[e_g, l_g]$.

Each SCC draws biomedical samples from individuals from opening time a_g until closing time $b_g = a_g + O_g$.

We now examine the samples. The maximum time allowed is from the moment a sample is drawn until its arrival at the Lab. Parameter T^g_{max} presents the shortest useful lifetime of the samples taken at SCC c_g.

Concerning logistic activities, τ_g denotes the time required for the truck driver to collect the samples at c_g by τ_0.

The limitation imposed by T^g_{max} may require visiting some SCCs several times during their opening hours. The **minimum number** of visits required by c_g can be estimated as $|P_g| = \left\lceil \frac{O_g}{\hat{T}_g} \right\rceil$, where P_g denotes the set of visits to perform, and $\hat{T}_g$ is the longest time allowed between two consecutive visits at c_g, assuming the samples are brought directly to the Lab. Given t_{g0}, the travelling time from c_g to the Lab, $\hat{T}_g$ can be computed as $\hat{T}_g = T^g_{max} - t_{g0} - \tau_g$. Note that, because no sample can stay at the SCC overnight, the last visit must be planned at every SCC c_g after closing time. The staff of that center define φ_g, which is the maximum amount of time they can wait for the last visit after closing.

Despite the previous estimates, the number of times that c_g must be visited depends on its opening and closing times and on the decisions made, particularly on the time at which c_g receives the first visit. Indeed, as soon as the first sample is drawn, it sets 1) the latest arrival time at the Lab and 2) the latest time at which c_g must be visited again. Similarly, when a second visit is planned (no later than the latest time set by the previous visit), a bound is set on the next visit, and so on, until the last visit is planned after the closing time of c_g.

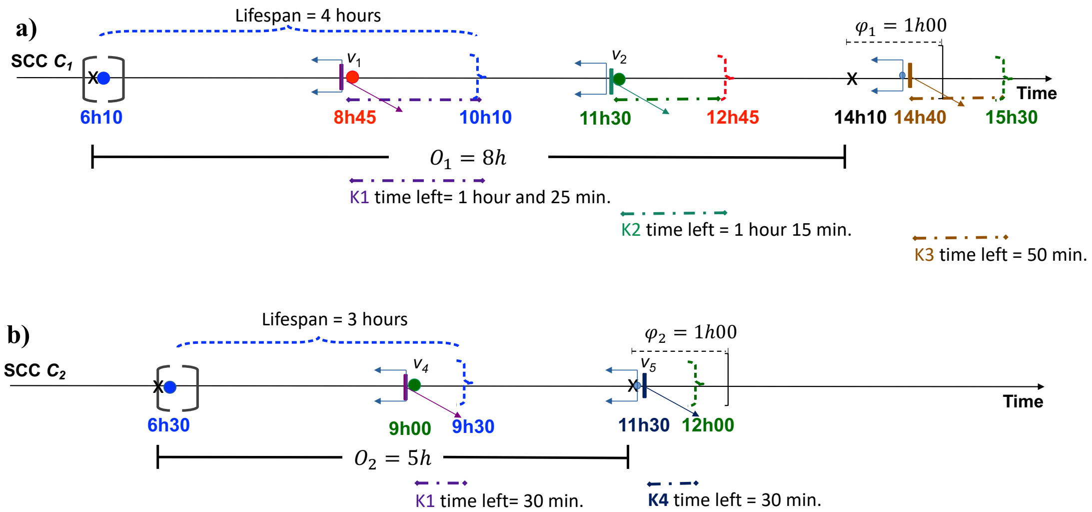

Figure 1: Feasible solution for BSTP. Part a) SCC c_1 and Part b) SCC c_2

To better illustrate the intertwined relationships between routes visiting the same SCC, consider the feasible solution for SCC c_1 in Figure 1, part a). In this example, c_1 has a maximum time T^g_{max} of 4 hours and opens at 6:10 am. The first visit v_1 is scheduled for 8:45 a.m. to obtain all accumulated samples. Hence, vehicle K_1 has until 10:10 a.m. (1:25 hours) to bring the samples to the Lab. Moreover, the timing decision of v_1 sets that the second visit at SCC c_1 (i.e., v_2) must be scheduled in time to bring samples to the Lab before 12:45 p.m.

Note that the bound on the latest time for the next visit is set based on the assumption that the vehicle will travel directly and immediately to the Lab right after leaving c_g. Therefore, this bound on the latest time imposes a restriction on the end of the route for the vehicle that performs the next visit. Consequently, a single visit at c_g affects both the rest of the first vehicle's route and the end of the route of the vehicle performing the next visit to c_g. For instance, if vehicle K_1 in Figure 1, part a) travels to visit SCC c_2 (Figure 1, part b), the routing time reduces by one hour and arrives at the Lab before 9:30 a.m. with respect to the time restrictions of c_2.

This planning process is challenging. Planning visits too close to their latest time reduces the flexibility to construct future routes. However, planning visits too early pulls the timing of future visits, which must be performed earlier, and eventually more visits than necessary might be performed at the same SCC. The BSTP decides the most efficient transportation to bring all samples collected at the SCCs in time to be analyzed at the Lab. Note that the managers force lifespans to be respected (set as a constraint) and seek to minimize transportation costs. To this end, the BSTP minimizes the route duration.

The next subsection proposes a formulation adapted from Anaya-Arenas et al. (2021) addressing this new method of considering the perishability of samples.

3.2 Mixed Integer Linear Programming Model formulation

The BSTP is modelled on an extended graph $G = (V, A)$ in which each SCC c_g is represented by $|P_g|$ nodes representing the visits required during the day. Without loss of generality, and to reduce symmetries, visits are labeled in such a manner that the first $|P_1|$ nodes in V correspond to the visits to c_1, $\{v_1, \dots, v_{|P_1|}\}$; then, $\{v_{|P_1|+1}, \dots, v_{|P_1|+|P_2|}\}$ are the ones visiting c_2, and so on. Specifically, defining the set of indexes is possible for visits at c_1 as $I_1 = \{1, 2, \dots, |P_1|\}$ and are analogous for a general SCC c_g $I_g = \left\{1 + \sum_{h=0}^{g-1} |P_h|, 2 + \sum_{h=0}^{g-1} |P_h|, \dots, |P_g| + \sum_{h=0}^{g-1} |P_h|\right\}$, where $|P_0|$ is set to 0.
Additionally, the graph's set of arcs can be specified as $A = \left\{(v_i, v_j) : v_i, v_j \in V,\ i \neq j,\ i, j \in \{0, \dots, |P|\}\right\}$ where each arc (v_i, v_j) is characterized by transportation time t_{ij}. As the graph models real routes, we assume that $t_{ij} \neq t_{ji}, \forall i, j \in P, i \neq j$. To present the mathematical formulation of the problem, we used the sets and parameters presented in Section 3.1. The sets, parameters, and decision variables of the program, along with their domains and meanings, are grouped in Table 1.

The notation $(A)_n$ is used to indicate the n-th element of the ordered set A. We also set a global maximum time $T_{max} = \max_{g \in [1, \dots, n]} T^g_{max}$, which is driven by the less restrictive SCC of the network. Given this, we can consider for each SCC the difference $\delta_g = T_{max} - T^g_{max}$, which is a quantity in the range $[0, T_{max}]$. The formulation is as follows:

Objective:

$$Min \quad \sum_{i=1}^{|P|} d_i \tag{1}$$

Sets

C	Specimen Collection Centers (SCC)
P	Visits scheduled for all SCCs
P_g	Visits required by the single SCC c_g
I_g	Indices for visits required by SCC c_g

Parameters

n	Total number of SCC in the network
O_g	Length of the collection period at SCC c_g
e_g	Earliest opening of SCC c_g
l_g	Latest opening of SCC c_g
φ_g	Maximum waiting time after shutting for the last pickup at SCC c_g
T^g_{max}	Maximum time limit within production and arrival to the Lab, for c_g
T_{max}	Maximum T^g_{max} over all SCC
δ_g	Difference among T_{max} and T^g_{max}
τ_g	Loading time at SCC c_g
τ_0	Unloading time at the Lab
t_{ij}	Transportation time between centers i and j
$\hat{T}_g$	Maximum time between two consecutive pickups at SCC c_g

Decision variables

x_{ij}	$\{0, 1\}$	is 1 if node i is visited immediately before node j, and 0 otherwise;
u_i	$\mathbb{R}^+$	time at which visit i is performed;
d_i	$\mathbb{R}^+$	duration of the route starting with visit i;
f_i	$\mathbb{R}^+$	longest remaining time at node i to complete the route;
a_g	$\mathbb{R}^+$	opening time of SCC c_g;
b_g	$\mathbb{R}^+$	closing time of SCC c_g

Table 1: Notation

Subject to:

$$\sum_{i=0}^{|P|} x_{ij} - \sum_{i=0}^{|P|} x_{ji} = 0; \qquad j = 0, ..., |P| \tag{2}$$

$$\sum_{i=0}^{|P|} x_{ij} = 1; \qquad j = 1, ..., |P| \tag{3}$$

$$e_g \leq a_g \leq l_g; \qquad g = 1, ..., n \tag{4}$$

$$a_g + O_g = b_g; \qquad g = 1, ..., n \tag{5}$$

$$u_j \geq u_i + \tau_i + t_{ij} - M(1 - x_{ij}); \qquad i = 0, ..., |P|;\ j = 1, ..., |P|;\ (i \neq j) \tag{6}$$

$$u_k \geq a_g; \qquad g = 1, ..., n;\ k = (I_g)_1 \tag{7}$$

$$u_k \geq u_{k-1}; \qquad g = 1, ..., n \text{ s.t. } |P_g| > 2;\ k \in I_g \text{ s.t. } (I_g)_1 < k \tag{8}$$

$$b_g \leq u_k \leq b_g + \varphi_g; \qquad g = 1, ..., n;\ k = (I_g)_{|P_g|} \tag{9}$$

$$T_{max} - f_i + M(1 - x_{i0}) \geq t_{i0} + \tau_i; \qquad i = 1, ..., |P| \tag{10}$$

$$f_j - f_i + M(1 - x_{ij}) \geq u_j - u_i; \qquad i, j = 1, ..., |P|\ (i \neq j) \tag{11}$$

$$f_i - (u_i - a_g) \geq \delta_g; \qquad g = 1, ..., n;\ i = (I_g)_1 \tag{12}$$

$$f_i - (u_i - u_{i-1}) \geq \delta_g; \qquad g = 1, ..., n;\ \text{ s.t. } |P_g| \geq 2;\ i \in I_g \text{ s.t. } i > (I_g)_1 \tag{13}$$

$$d_i \geq T_{max} - f_i + t_{0i} + \tau_0 - M(1 - x_{0i}); \qquad i = 1, ..., |P| \tag{14}$$

$$u_i, f_i, a_g, b_g, d_i \in \mathbb{R}^+; \qquad i = 1, ..., |P|;\ g = 1, ..., n \tag{15}$$

$$x_{ij} \in \{0, 1\}; \qquad i, j = 1, ..., |P|\ (i \neq j) \tag{16}$$

The objective function (1) aims to minimize the total duration of all routes in the solution. This objective function assumes that transportation is routed by third-party logistics such that there is no fixed cost of vehicles (Anaya-Arenas et al., 2016). Constraint (2) ensures flow conservation in every node of the graph, whereas constraint (3) ensures that all visits are performed by requiring all nodes to have a predecessor in the solution. Constraints (4) and (5) control the flexible opening window of the SCCs; in particular, Constraint (4) ensures that each SCC c_g opening time is within its time opening window $[e_g, l_g]$, and (5) defines the closing hour of the SCC.

Constraints (6)–(9) define the time at which each visit is performed. Constraint (6) sets u_j, which is the time of each visit j, to u_i, which is the time of the visit to its predecessor in the route (i such that $x_{ij} = 1$), plus the loading time at i (τ_i) and the travelling time from i to j (t_{ij}). Constraint (6) also forces sub-tour elimination. Constraint (7) states that no visit can be performed at any SCC c_g before its opening time. Constraint (8) requires visits to be performed in chronological order and avoids visit symmetries. Finally, Constraints (9) states that the last visit at each SCC g is performed after its closing hour, b_g, within φ_g units of time, with $k = \sum_{h=1}^{g} |P_h|$, or, equivalently, $k = (I_g)_{|P_g|}$.

Constraints (10)–(14) also relate to time but model time as a *resource* consumed during the route. Constraint (10) states that every visit i, which is performed immediately before the vehicle returns to the Lab (that is, $x_{i0} = 1$), must have its resource variable reduced by the collection and transportation times between i and 0. Constraint (11) relates the resource and visit time variables for any pair of consecutive nodes (i, j). Indeed, if arc (i, j) is included in the route (that is, $x_{ij} = 1$), then the difference in time consumption must match the time variables of the visits. Constraints (12) and (13) ensure that the samples do not perish by requiring that the sum of the time the samples stay at the SCC, plus the time required to bring them to the Lab, do not exceed the lifespan of the samples. Constraint (14) defines the duration of a route starting at node i. Indeed, if arc $(0, i)$ is in the solution, a route starting from node i exists, and its duration equals the travel time t_{0i} plus the unloading time at the laboratory τ_0 and the time consumed from i back to the Lab, that is(T_{max} - f_i). If no route starts from i, i.e., $x_{0i} = 0$, d_i is set to zero owing to the objective function structure.

Finally, Constraints (15) and (16) state the domains of the decision variables, as they are reported in Table 1.

4 The Matheuristic

To address the challenges raised by the BSTP, we conceived a matheuristic that combines two ideas. The first one comprises an iterative temporal-decomposition algorithm. Inspired by the rolling horizon (RH) approach, the algorithm separates the problem at each iteration into a different set of smaller, easier to solve subproblems that are solved using the mathematical formulation proposed in Section 3.2. The second idea lies in the use of some *Fix-&-Optimize* (F&O) strategies that speed up the search while guaranteeing an effective exploration of the solution space associated with each subproblem. Figure 2 depicts the algorithmic frame of the proposed matheuristic.

Subsections 4.1 and 4.2 present the "outer loop", which divides the problem into several subproblems. They explain how the RH scheme is applied to the BSTP and how an iterative algorithm produces different time decompositions, respectively. Then, the "inner loop" is explained, which focuses on solving each subproblem and improving the solution. Subsection 4.3 details how the mathematical formulation extends to solve subproblems within the RH scheme, whereas Subsection 4.4 describes the considered F&O techniques. Finally, Subsection 4.5 overviews the complete matheuristic.

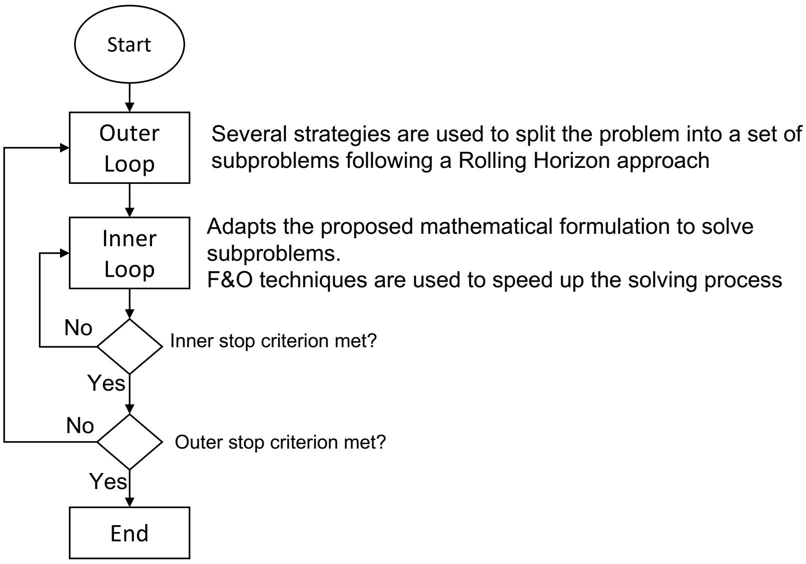

Figure 2: Algorithmic structure of the matheuristic

4.1 A Rolling Horizon scheme to solve the BSTP

Rolling horizon techniques are commonly used to approach complex problems that can are decomposable over time. In our scheme, decomposition is performed by choosing specific times, referred to as "cutting points", which divide the planning horizon [0,T] into a set of N consecutive *periods*. A subset of visits to perform (to SCCs) is then assigned to each period such that each period becomes a subproblem.

Subproblems are solved in a sequential manner, starting with the first one and proceeding to subproblem $|N|$, in such a manner that when solving subproblem κ, information on the previous subproblems is known. In addition, when solving subproblem κ, *looking-ahead* and simultaneously solving one or more visits from forthcoming subproblems may be considered.

Given a subproblem or period κ, the following periods or subproblems can be formalized:

- *Frozen Period* corresponds to subproblems already solved. The problem variables that fall within this range are known and can be set (or not, as Section 4.3 explains) to the previously computed values ;
- *Central or current Period* states the current subproblem κ to optimize. Variables associated with subproblem κ are free and must be set; and
- *Look-ahead Period* in which variables associated with forthcoming periods must be considered to guarantee the feasibility of the entire solution.

In the RH scheme, the *central period* κ becomes a *frozen period* when solving subproblem $\kappa + 1$, whereas the *look-ahead* period becomes the *central period.* This process repeats until the entire planning horizon is covered.

The next section explains how a BTSP is split into consecutive periods to apply an RH scheme.

4.2 An iterative algorithm to generate alternative decompositions to the BSTP

The application of an RH approach is not straightforward for the BSTP. Unlike classical RH applications in which a given decomposition is given by the nature of the problem or set arbitrarily by the users. In our case, neither clear indications on the adequate strategy to "split" the original problem, the number of sub-problems to create, nor their length/size exist. Indeed, the time at which visits will be done, which depends on the actual routes and previous visit decisions, is unknown a priori such that evident manners to set cutting points or to assign visits to the periods do not exist. Therefore, instead of producing a single decomposition, we elected to conceive an algorithm that adapts the time decomposition at each iteration to produce different sets of subproblems. Starting from an initial solution, the first-integer feasible solution produced by solving the formulation given in Section 3.2, the algorithm encompasses four decomposition methods, referred to as *DM1* to *DM4*, that are alternatingly used until reaching the stopping criterion of the algorithm. The four decomposition methods are as follows:

- *DM1* - Parametric decomposition scheme. This is controlled by three parameters: R, α, and β. R sets the number of visits to be considered in the central period; thus, fixing its length and size. α sets the length of the frozen period that contains the earliest $\lceil \alpha \times R \rceil$ visits. Finally, β defines the look-ahead horizon by setting the number $\lceil \beta \times R \rceil$ for future visits to be included in the subproblem. The values of the parameters are adjusted from one iteration to the next to produce different subproblems.
- *DM2* - Period extension. Each time a subproblem is solved in the inner loop, we evaluate whether the current solution can be locally improved by including visits that are currently planned in the look-ahead period in the central period. If true, the extended cutting points are recorded. When the DM2 decomposition method is called, the best cutting points in memory are used to define the new decomposition.
- *DM3* - Estimated times for route ends. This method tracks the moments at which routes end in the previously explored solutions and proposes cutting points that match the route ending times with the highest frequency.
- *DM4* - Fixed number of periods. This simply splits the entire horizon into the required number of periods. All the periods have the same length (number of visits).

Considering the specific characteristics of each decomposition method, we decided to use them in the following order. The first decomposition was provided by a parametric scheme. *DM1* is used while it improves the best solution found in the previous iteration. In addition, the parameters of *DM1* are adjusted such that if the produced subproblems are too difficult to solve (i.e., they require too much time), then R is reduced; otherwise, α is reduced and, β is increased, aiming to provide more freedom to variables in the problem and to look ahead farther, respectively. *DM2* is then applied using the information in memory (best cutting points thus far) to generate a new decomposition. Then, assuming that several solutions were explored, *DM3*, the estimated times for end-of-routes method, is applied to produce another decomposition of the problem. The last method, *DM4*, is used to hard diversify the search because it produces subproblems that can significantly differ from the ones explored thus far. Figure 3 at the end of this section illustrates how the different splitting strategies are used within the search algorithm. Note that the decomposition approach can produce different sequences of subproblems by changing the initial sequence of the visits to perform. This is a diversification technique of our algorithm.

4.3 Adapting the BSTP formulation to solve periods within the Rolling Horizon scheme

The interdependency between routes makes it inappropriate to solve a part of the problem without considering those previous and forthcoming. Moreover, in a given decomposition, a route may span two or more subproblems. Therefore, when solving a subproblem, extending the original formulation proposed in Section 3.2 is necessary to ensure the feasibility and continuity of the routes over the entire horizon.

The first modification to be applied to the original formulation concerns the information originating from the frozen period. Although was previously mentioned that the frozen variables are set to the values found when solving previous periods, in practice, we relax part of this information to focus on the initial arcs of the routes. Hence, when solving period κ, we force all existing routes to start at the same nodes that were selected as starting nodes in the previous period $\kappa-1$ while leaving complete freedom to the construction of new routes. This choice ensures the feasibility between *frozen* and *central* periods because we allow changes in the frozen solution, if necessary, in terms of the timing and parts of the routes (beyond the first visit). Formally, the *freezing strategy* can be expressed as Equation (17):

$$x_{0,i}^{\kappa} \geq x_{0,i}^{\kappa-1} \quad \forall\, i \tag{17}$$

where subscripts $\kappa-1$ and κ indicate the period or subproblem solved and to solve, respectively. Although this strategy progressively increases the computational time required to solve subproblems, it exploits that the first visit in a route plays a crucial role in defining the structure of the route while leaving the solver freedom to produce feasible solutions.

The second modification concerns the "future" visits that should be completed in future periods. Equation (18) provides a lower bound on the visit time at node k to avoid performing the last visit at SCC g included in the sub-problem too early (visit k, $k = (\tilde{I}_g)_{|\tilde{P}_g|}$) such that completing this partial solution and obtaining a feasible solution to the entire problem is always possible. That sis, all feasible solutions of the subproblem represent partial feasible solutions for the entire problem.

$$u_k \geq b_g - (|P_g| - |\tilde{P}_g|)\hat{T}_g; \quad g = 1, ..., \tilde{n};\ k = (\tilde{I}_g)_{|\tilde{P}_g|} \tag{18}$$

The complete mathematical formulation for solving each subproblem is provided in the Appendix.

4.4 Fix–and–Optimize (F&O) strategies

To reduce the computational time required to solve the BSTP formulation, we propose F&O strategies to reduce the space of solutions to explore in the inner loop and use them as intensification mechanisms. We implemented two F&O strategies, referred to as *Keep groups* and *Change arcs*.

Given a current solution s, the *Keep groups* strategy imposes that at least a given number of the visits must be maintained on the same route in the new solution $\bar{s}$, although their order and visiting times may change. Subsequently, we guarantee that, from one solution to the next, the composition of the routes will differ (in sequence) while maintaining the structure (set of visits), reducing the computational time. Formally, this strategy can be expressed by the following constraint:

$$\sum_{i \in r} \sum_{j \in r\, j \neq i} x_{ij} \geq |r| - 1 \quad \forall\, route\, r \tag{19}$$

where r represents the route, i and $j \in r$ the nodes to be visited by route r, and $|r|$ the cardinality of the route (i.e., the number of centers visited by route r).

The *Change arcs* strategy, which was inspired by probabilistic approaches such as *granular tabu search*, requires that, in the new solution, at least a certain number of arcs Δ are taken from a subset of "promising arcs" $\bar{A}$. Subset $\bar{A}$ is built and managed as the research advances, keeping in memory the number of times that each arc is inserted into a solution. Subsequently, the solver has limited freedom in the choice of the arcs to use, controlled by parameter Δ. This type of strategy can be formalized using the following constraint:

$$\sum_{(i,j)\in\bar{A}} (1 - x_{ij}) + \sum_{(i,j)\notin\bar{A}} x_{ij} \geq \Delta \tag{20}$$

where $\bar{A} = \{(i,j) : \bar{x}_{ij} = 1\}$ is the set of promising arcs.

The strategies described show two different levels of intensification, which are complementary. Indeed, while the *Keep groups* strategy seeks to group visits without regarding the structure of the routes, the *Change arcs* strategy intends to encourage the presence of parts of routes that were deemed appealing. Therefore, their use must be carefully chosen. The following principles guided their implementation in the inner loop.

- The first iteration of the inner loop runs without a strategy;
- Any time a new best solution is found, the *Keep groups* strategy is used; otherwise use *Change arcs*;
- Stop if it_M consecutive iterations are executed and no solution improvement is reached.

Choosing a low number of iterations without improvement as a stop criterion guarantees that, given a limit on the computational time, the matheuristic explores more partitions of the problem, but at the price of a less thorough exploration of each.

4.5 The complete Matheuristic

Figure 3 illustrates the algorithmic structure of the proposed matheuristic. Using an initial solution, decomposition method DM1 is applied using the initial parameters (R_0, α_0, β_0) to produce an initial problem decomposition that is transferred to the inner loop (the rolling horizon RH-solving method). The RH is then used to solve the set of subproblems yielded by the first decomposition, first without any F&O strategy, and, if no improvement is reached with respect to the best solution found thus far (in this case, the initial solution), then the F&O strategies are used. The algorithm iterates between the outer and inner loops (i.e., the left and right parts of the figure, respectively), producing different decompositions of the problem until the stop criterion is reached; in our case, the total number of outer-loop iterations I_M.

5 Results

The aim of this section is twofold. First, it assesses the performance of the proposed matheuristic in terms of the quality of the solutions it produces and required computational time. Second, it analyzes the results produced by the matheuristic to highlight the specific contribution and added value of the different mechanisms and algorithmic strategies in the proposed matheuristic.

5.1 Description of the test instances and setting of the algorithm's parameters

We first describe the characteristics of the 32 instances used in our numerical experiments. Instances were categorized into 17 *medium* and 15 *large* size instances, according to the cardinality

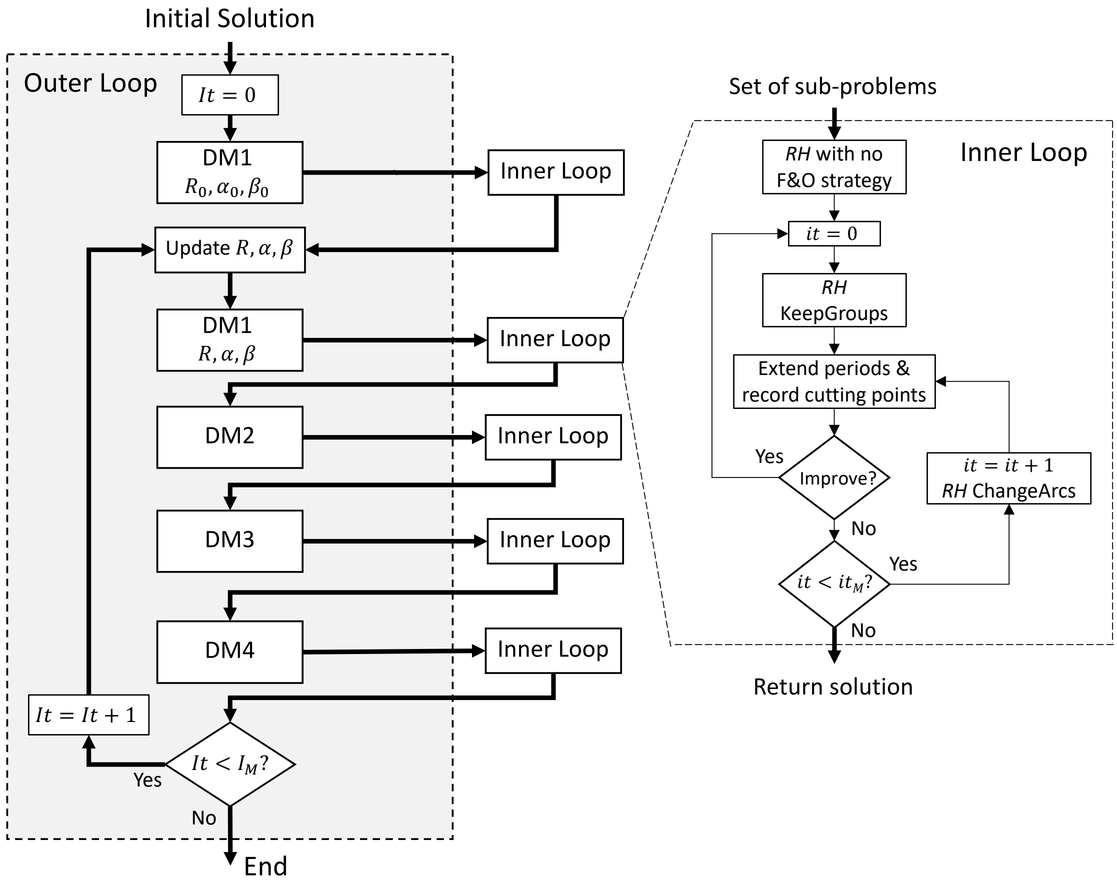

Figure 3: Algorithmic structure of the matheuristic

of the set of collection centers to deserve $|SCC|$ and the number of visits to perform $|P|$. In the group of *medium* instances, the $|SCC|$ ranged from 4 to 24, which corresponds to the number of visits $|P|$ varying from 11 to 28. To measure the interdependency level of an instance, we calculated the synchronization percentage as the percentage of SCCs in an instance that required two or more visits in a single day. Medium instances present a synchronization percentage between 23 and 100%, with nine instances requiring synchronization above 80%. In the *large* size instances, the $|SCC|$ ranged from 11 to 50, and the number of visits $|P|$ varied between 29 and 74. In large instances, the synchronization percentage increased from 63% to 100%, with 11 instances requesting a synchronization above 80%.

Moreover, the demographic and topological aspects are as relevant as the size of the instance to understand the richness and diversity of the testbed. Indeed, some instances consider vast territories with a low population density and a rather light network of routes. We refer to the 17 instances matching this description as "rural" (11 medium and six large instances). The 15 remaining instances are much denser; thus, the distances between SCCs are shorter, and the number of arcs connecting them is much higher. We refer to these as "urban" instances (six medium and nine large instances).

The proposed matheuristic has some parameters that must be set adequately. To this end, preliminary tests were conducted to help us elect parameter values. We arbitrarily fixed $it_M = 2$ and $I_M = 7$ to ensure that various time decompositions were considered for each instance. As mentioned in Section 4.4, the number of inner iterations was dynamically addressed by limiting the number of consecutive iterations without improvement to it_M.

We also limited the computational time required to solve each subproblem to 90 s, which was sufficiently large to solve most of the subproblems.

Based on the preliminary experiments, we set the initial values for the parameters of the outer loop to $(R_0, \alpha_0, \beta_0) = (7, 0.5, 0.5)$.

Then, for subsequent iterations, these values were updated as follows. If the time required to solve the complete set of subproblems was shorter than a given threshold (350 s), then α_0 was decreased and β_0 increased.

Otherwise, we assumed that the decomposition method produced subproblems requiring too much time to solve, and, to reduce the computational time, R was decreased to $R - 2$ visits. Finally, we set parameter Δ, which defines the number of arcs to be selected from the promising arc set $\bar{A}$ in the inner loop, to $\Delta = \lceil 0.8|\bar{A}| \rceil$.

All tests were executed on a multi-user server with 64GB of RAM and am Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz, with 8GB of RAM. The MILP formulation was solved using CPLEX 12.8.

5.2 Numerical results

To assess the performance of the proposed matheuristic, we used the best results produced by the mathematical formulation presented in Section 3.2 as a baseline, referred to as MILP, within a time limit of 36 000 s (10h). Table 2 lists the results. The leftmost part lists the instances and describes their main characteristics, starting with the number of collection centers $|SCC|$ and the number of visits to perform $|P|$. The results produced by the mathematical formulation are then reported, including the value of the objective function (OF), computational time in seconds (column *sec.*), and optimality gap reported by the solver (GAP). Note that in several cases, the optimality gap was not closed before exhausting the allotted computational time (36 000 s). In other cases, however, the computer memory limit ($8GB$) was reached, stopping the search. An asterisk ($*$) identifies the cases for which the search aborted in the computational time column.

The last two columns in Table 2 report the matheuristic results. Column %MILP yields the difference in percentage between the best solutions produced by the two methods computed as %MILP= $(OF_{Matheu} - OF_{MILP})/OF_{MILP}$. Therefore, the negative values of %MILP indicate that the matheuristic produced a solution better than that of MILP. Finally, column *sec.* reports the computational time required by the matheuristic to solve each instance.

Let us first look at the results produced by the MILP for the *medium* size instances (instances 1 to 17). Table 2 confirms the difficulty of solving this problem. Indeed, CPLEX was able prove optimality only for instances 2, 3, and 13, and, in the case of instance 13, doing such required more than 26 000 s. For the five cases in which the allotted computational time was exhausted, the optimality gaps were greater than 40%. Such large optimality gaps do not allow us to confirm the quality of the best solution found thus far by the solver. Finally, in nine cases, the search was aborted after the computer's memory limit was reached. The heuristic produced the same objective value as the MILP in 10 of 17 instances and improved the best solution of the MILP in 4 of 17 instances, although the MILP performed better in three cases. Per the computational time, most instances required between 1500 and 3300 s, with an average computational time of 2167 s, confirming that the heuristic was able to reach solutions as good as CPLEX in less than 40 min.

If we examine the results produced for *large* size instances (instances 18 to 32), CPLEX produced solutions that, in the best case, showed an optimality gap of 71%. In 11 cases, the

				MILP			Matheuristic	
Ins.	$\lvert SCC \rvert$	$\lvert P \rvert$	**Dem**	**OF**	**sec.**	**Gap**	**%MILP**	**sec.**
1	4	17	R	1870	*	100%	-0.05%	1535
2	6	11	R	502	5	0%	0.00%	59
3	7	14	R	1137	467	0%	0.00%	279
4	8	24	R	2606	*	100%	0.49%	2632
5	9	16	R	1252	36000	40%	0.00%	981
6	9	26	R	1847	36000	100%	0.00%	2901
7	12	18	R	1407	*	77%	0.00%	1973
8	17	24	R	2009	*	100%	-0.15%	2906
9	19	22	R	1581	*	90%	0.00%	2961
10	23	27	R	2020	*	89%	0.00%	2838
11	24	28	R	2019	*	91%	0.00%	2044
12	8	15	U	516	36000	49%	0.00%	1602
13	9	14	U	439	26112	0%	0.00%	1570
14	10	24	U	812	36000	100%	-1.48%	3195
15	11	25	U	811	*	100%	2.10%	2875
16	12	26	U	913	*	100%	-0.66%	3297
17	13	28	U	908	36000	100%	1.98%	3188
							0.13%	2167
18	11	29	R	2490	*	100%	-0.12%	2768
19	14	36	R	2453	*	100%	-2.60%	3751
20	26	31	R	2193	*	93%	0.00%	3292
21	40	63	R	3987	36000	100%	4.82%	4126
22	46	70	R	5017	36000	100%	0.79%	4750
23	50	74	R	5289	*	100%	1.68%	4002
24	17	29	U	1905	*	89%	-6.51%	2900
25	17	33	U	924	*	100%	-1.19%	3301
26	18	35	U	1860	*	100%	0.16%	3708
27	19	33	U	1898	*	100%	-0.16%	3646
28	19	33	U	1898	*	90%	-0.84%	3500
29	19	35	U	2208	36000	94%	-1.90%	3422
30	27	32	U	3095	*	71%	-0.85%	3095
31	28	33	U	3131	*	76%	0.00%	3332
32	35	40	U	3746	36000	87%	0.27%	3960
							-0.43%	3570

Table 2: Numerical results produced for the *medium* and *large* size instances

search tree exhausted the memory of the available computer, aborting the search. The matheuristic improved the results of the MILP in 8 of 15 cases and produced the same objective value in 2 more cases, whereas the MILP was better in five cases. The matheuristic produced an average improvement of 0.43% over the *large* size instances, which is encouraging when considering the computational time required to reach these solutions. Indeed, the computational times required by the heuristic to solve *large* instances remained within the same order of magnitude as those for *medium* instances, ranging from 2 768 to 4 750 s (46 to 80 min). This confirms the good scalability of the matheuristic and its potential for efficiently handling even larger instances.

Previous experiments demonstrated the effectiveness and efficiency of the proposed matheuristic. The next section analyzes how the different mechanisms and search strategies that form the matheuristic contribute to its performance. We begin with inner-loop strategies. As described in Section 4.4, two fixed & optimization strategies were proposed in the inner loop: the *Keep Groups*, which plays a route intensification role; and the *Change arcs*, which aims to focus on the search considering promising solutions. To assess the extent to which these strategies work in an intertwined manner, the left part of Figure 4 shows the usage of the two strategies in the inner loop, whereas the right part shows the number of times that the best solution was produced using each strategy. Because the considered instances show distinct topological natures (i.e., rural vs. urban), our analysis explicitly considered this.

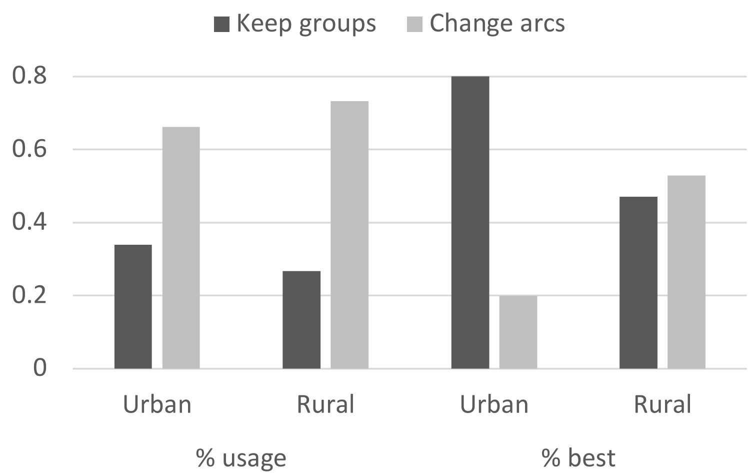

Figure 4: Portion of times the algorithm uses different inner loop strategies, among rural and urban instances

In our case, the left part of Figure 4 (over title %usage) shows that although the *Change arcs* strategy is used more frequently than the *Keep groups* strategy, the latter is still used more than 1/3 of the times. Evidently, the use of these strategies depends on the type of instance. Denser urban instances perform fewer *Change arcs* iterations (approximately 66% of the total) than rural instances (74% of the total iterations). This suggests that the dynamic mechanism allowing to switch between them works adequately. Moreover, the right part of Figure 4 (over title %best) reports the portion of times that the matheuristic produced its best solution during an iteration using the *Keep groups* or the *Change arcs* strategy. In the case of urban instances, the best solution was produced in 80% of the cases using the *Keep groups* strategy, and this percentage reduces to only 47% when solving rural instances. This makes sense; as the routes produced for urban instances have more visits, *Keep groups* force the solver to focus on the route configuration to determine the best timing for the visits. The complementarity of the proposed strategies and their contribution to the effectiveness of the matheuristic are confirmed.

Let us now focus on assessing the extent to which the proposed outer-loop (i.e., diversification) strategies can split the problem into effective subproblems. To this end, we identified the problem decomposition for each of the 32 instances for which the best solution was reached. Figure 5 reports the number of times each problem decomposition method led to the best solution for rural and urban instances.

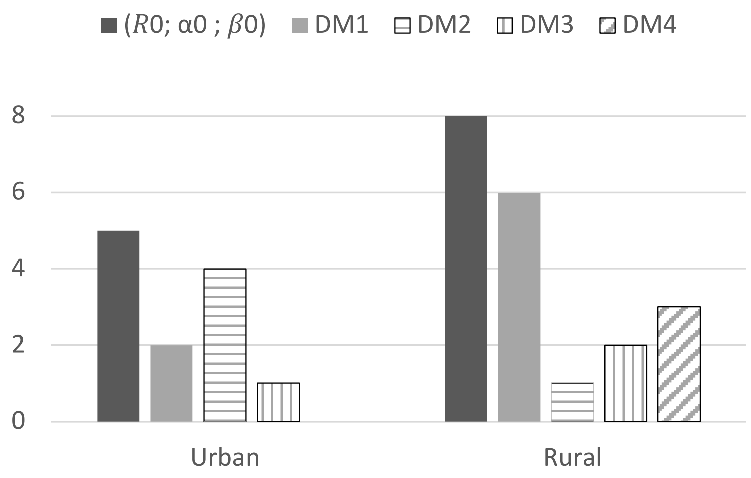

Figure 5: Number of times each problem decomposition method led to the best solution for rural and urban instances

The initial problem-splitting with parameters ($R = 7, \alpha = 0.5, \beta = 0.5$) produced the largest number of best solutions (13), confirming that the empirically chosen values were quite effective. Note that mechanism $DM1$, which updates the values of parameters (R, α, β), allows for problem partitions that led to eight best solutions, whereas the problem decomposition produced by $DM2$, the *period extension* method, allowed five best solutions to be reached. Finally, the last two splitting approaches, $DM3$ and $DM4$, contributed to six best solutions, although $DM4$ did not produce any best solution for urban instances. We conclude that all the strategies used contributed to the effectiveness of the matheuristic, and that their different principles enhance its robustness.

Finally, we inquired about the time at which the best solutions were found. In an attempt to demonstrate the value of the time invested and the strategies used during the search, we normalized the computational time across the instances to present in Figure 6: the number of times that the best solution was reached during the first 20% of the total computational time, between the 20% and the 40% of the total computational time, etc.

The largest number of best solutions (20) was produced during the first part of the search within 40% of the time, which is consistent with the success of the initial splitting strategy, as previously discussed. Nevertheless, up to 10 best solutions were reached during the 40% to 80% quartile of the total computation time. Unsurprisingly, only two best solutions were obtained at the end of the experiment.

To summarize, the proposed matheuristic proposes an adequate combination of strategies and mechanisms that allow it to effectively tackle the difficulties and challenges raised by the BSTP, as the decomposition is not straightforward. In particular, the iterative approach that exploits

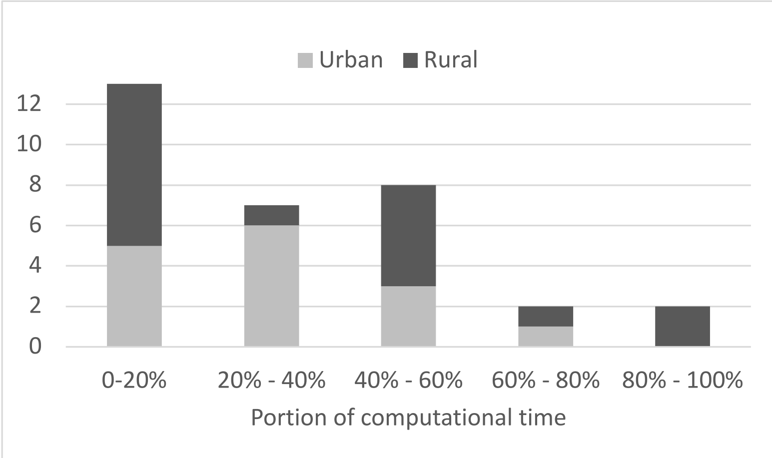

Figure 6: Number of best solutions produced within each quartile of the computational time

different time decompositions produced encouraging results and reveals is a promising approach for addressing routing problems with dependency or synchronization constraints between routes.

6 Conclusions

This paper presents an efficient matheuristic for solving the complex routing problem of transporting products with short lifespans and in which visits to customers are interdependent. The real challenges of the biomedical sample transportation problem (BSTP) inspired this study, which is a present problem in Quebec (Canada). The problem studied relates to VRPTW-Syn and blood transportation, but with a strong interdependence in the decisions concerning visits to customers and route schedules. It extends previous formulations for the BSTP by realistically considering the lifespan of samples.

To tackle the complex MILP formulation and shortcomings of commercial solvers, this paper proposes an efficient matheuristic that combines an iterative time-decomposition approach and two Fix-&-Optimize (F&O) techniques. The iterative matheuristic proposes a novel method of reducing interdependency by applying different strategies to divide the problem into shorter horizons with different visit structures. By solving these subproblems, the algorithm allows the solver to efficiently explore the solution space. Four decomposition mechanisms were proposed and tested, which provided good performances and contributed to the search. Moreover, the two F&O techniques act as intensification mechanisms, thereby increasing the efficiency of the algorithm. We tested the matheuristic on real-life instances with up to 50 customers, 74 visits, and a requested synchronization average of 80%, which is much larger than the synchronization percentage requested in previous studies. The matheuristic provides good quality solutions, as good or better than CPLEX, but in only a fraction of the time, making it suitable for implementation in a decision support tool for planners in healthcare logistics.

Despite these encouraging results, additional research is required to assess the effectiveness of the iterative time-decomposition scheme on other problems and instances. In addition, we believe that the use of a heuristic approach rather than the adapted mathematical formulation for solving the subproblems generated by the time decomposition would allow us to solve larger subproblems, thus improving the effectiveness of the method.

Acknowledgments

The authors thank the Natural Sciences and Engineering Research Council (NSERC) of Canada for its financial support under Grants no RGPIN-2018-04609 and the *Fonds de Recherche du Quebec en Nature et Techonologies* under grant 2021-NC-282544.

References

Afifi, S., Dang, D. C., & Moukrim, A. (2016). Heuristic solutions for the vehicle routing problem with time windows and synchronized visits. *Optimization Letters*, *10*(3), 511–525.

Ait Haddadene, S. R., Labadie, N., & Prodhon, C. (2016). A GRASP × ILS for the vehicle routing problem with time windows, synchronization and precedence constraints. *Expert Systems with Applications*, *66*, 1339–1351.

Anaya-Arenas, A. M., Chabot, T., Renaud, J., & Ruiz, A. (2016). Biomedical sample transportation in the province of Quebec: A case study. *International Journal of Production Research*, *54*(2), 602–615.

Anaya-Arenas, A. M., Prodhon, C., Renaud, J., & Ruiz, A. (2021). An iterated local search for the biomedical sample transportation problem with multiple and interdependent pickups. *Journal of the Operational Research Society*, *72*(2), 367–382.

Araújo, A. M., Santos, D., Marques, I., & Barbosa-Povoa, A. (2020). Blood supply chain: a two-stage approach for tactical and operational planning. *OR Spectrum*, *42*(4), 1023–1053.

Baş, S., Carello, G., Lanzarone, E., Ocak, Z., & Yalçındağ, S. (2016). Management of blood donation system: Literature review and research perspectives. *Springer Proceedings in Mathematics and Statistics*, *169*, 121–132. doi: 10.1007/978-3-319-35132-2_12

Baş, S., Carello, G., Lanzarone, E., & Yalçındağ, S. (2018). An appointment scheduling framework to balance the production of blood units from donation. *European Journal of Operational Research*, *265*(3), 1124–1143.

Brailsford, S., & Vissers, J. (2011). OR in healthcare: A European perspective. *European Journal of Operational Research*, *212*(2), 223–234.

Bredström, D., & Rönnqvist, M. (2008). Combined vehicle routing and scheduling with temporal precedence and synchronization constraints. *European Journal of Operational Research*, *191*(1), 19–31.

Decerle, J., Grunder, O., Hajjam El Hassani, A., & Barakat, O. (2018). A memetic algorithm for a home health care routing and scheduling problem. *Operations Research for Health Care*, *16*, 59–71.

Doerner, K. F., Gronalt, M., Hartl, R. F., Kiechle, G., & Reimann, M. (2008). Exact and heuristic algorithms for the vehicle routing problem with multiple interdependent time windows. *Computers and Operations Research*, *35*(9), 3034–3048.

Dohn, A., Rasmussen, M. S., & Larsen, J. (2011). The vehicle routing problem with time windows and temporal dependencies. In *Networks.*

Drexl, M. (2012). Synchronization in Vehicle Routing—A Survey of VRPs with Multiple Synchronization Constraints. *Transportation Science*, *46*(3), 297–316.

Elalouf, A., Tsadikovich, D., & Hovav, S. (2018). Optimization of blood sample collection with timing and quality constraints. *International Transactions in Operational Research*, *25*(1), 191–214.

Euchi, J., Zidi, S., & Laouamer, L. (2021). A new distributed optimization approach for home healthcare routing and scheduling problem. *Decision Science Letters*, *10*(3), 217–230.

Frifita, S., & Masmoudi, M. (2020). VNS methods for home care routing and scheduling problem with temporal dependencies, and multiple structures and specialties. *International Transactions in Operational Research*, *27*(1), 291–313.

Ghandforoush, P., & Sen, T. K. (2010). A DSS to manage platelet production supply chain for regional blood centers. *Decision Support Systems*, *50*(1), 32–42.

Kergosien, Y., Ruiz, A., & Soriano, P. (2014). A Routing Problem for Medical Test Sample Collection in Home Health Care Services. In *Springer proceedings in mathematics and statistics* (Vol. 61, pp. 29–46).

Liu, R., Xie, X., Augusto, V., & Rodriguez, C. (2013). Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care. *European Journal of Operational Research*, *230*(3), 475–486.

Melachrinoudis, E., Ilhan, A. B., & Min, H. (2007). A dial-a-ride problem for client transportation in a health-care organization. *Computers and Operations Research*, *34*(3), 742–759.

Mobasher, A., Ekici, A., & Özener, O. Ö. (2015). Coordinating collection and appointment scheduling operations at the blood donation sites. *Computers & Industrial Engineering*, *87*, 260–266.

Naji-Azimi, Z., Salari, M., Renaud, J., & Ruiz, A. (2016). A practical vehicle routing problem with desynchronized arrivals to depot. *European Journal of Operational Research*, *255*(1), 58–67.

Osorio, A. F., Brailsford, S. C., & Smith, H. K. (2015). A structured review of quantitative models in the blood supply chain: A taxonomic framework for decision-making. *International Journal of Production Research*, *53*(24), 7191–7212.

Pirabán, A., Guerrero, W. J., & Labadie, N. (2019). Survey on blood supply chain management: Models and methods. *Computers and Operations Research*, *112*.

Rousseau, L.-M., Gendreau, M., & Pesant, G. (2013). The synchronized dynamic vehicle dispatching problem. *INFOR: Information Systems and Operational Research*, *51*(2), 76–83.

Yousefi Nejad Attari, M., Pasandideh, S. H. R., & Akhavan Niaki, S. T. (2019). A hybrid robust stochastic programming for a bi-objective blood collection facilities problem (Case study: Iranian blood transfusion network). *Journal of Industrial and Production Engineering*, *36*(3), 154–167.

Yücel, E., Salman, F. S., Gel, E. S., Örmeci, E. L., & Gel, A. (2013). Optimizing specimen collection for processing in clinical testing laboratories. *European Journal of Operational Research*, *227*(3), 503–514.

Zabinsky, Z. B., Dulyakupt, P., Zangeneh-Khamooshi, S., Xiao, C., Zhang, P., Kiatsupaibul, S., & Heim, J. A. (2020). Optimal collection of medical specimens and delivery to central laboratory. *Annals of Operations Research*, *287*(1), 537–564.

Zahiri, B., Torabi, S. A., Mohammadi, M., & Aghabegloo, M. (2018). A multi-stage stochastic programming approach for blood supply chain planning. *Computers and Industrial Engineering*, *122*(May), 1–14.

Şahinyazan, F. G., Kara, B. Y., & Taner, M. R. (2015). Selective vehicle routing for a mobile blood donation system. *European Journal of Operational Research*, *245*(1), 22–34.

Appendix

This appendix is devoted to the mathematical formulation that must be used when dealing with only a portion of the full problem, a *sub-problem*.
Given a network, consider the subproblem given by the pickup requests that must be performed in the morning. When excluding the second part of the day, a new problem is built that cannot be solved as if a second part of the day was nonexistent. Thus, an enriched formulation is required when coping with subproblems.

Following the same pattern as in Section 3, a formalization of a subproblem is first given before the mathematical program formulation.

Formalization of sub-problems

As anticipated, when excluding a part of the instance, not all requests of the original problem are contained. To address this, we can redefine all sets described in Section 3, those marked with a tilde, to state that they are related to this subproblem.
In particular,

$$\tilde{C} = \{c_1, c_2, \dots, c_{\tilde{n}}\} \tag{21}$$

is the set of SCCs with at least one request included in the subproblem.
Moreover,

$$\tilde{P}_g \subset P_g \tag{22}$$

is the set of requests for SCC g that are part of the subproblem.
Given this, it suffices to define

$$\tilde{P} = \bigcup_g \tilde{P}_g \tag{23}$$

to obtain the set of all requests related to the considered Lab.
Following the same reasoning, it is possible to define:

$$\tilde{V} = \{v_0, v_1, \dots, v_{|\tilde{P}|}\} \tag{24}$$

$$\tilde{A} = \big\{(v_i, v_j) : v_i, v_j \in V,\ i \neq j,\ i, j \in \{0, \dots, |\tilde{P}|\}\big\} \tag{25}$$

$$\tilde{I}_g = \big\{1 + \sum_{h=0}^{g-1} |\tilde{P}_h|,\ 2 + \sum_{h=0}^{g-1} |\tilde{P}_h|, \dots,\ |\tilde{P}_g| + \sum_{h=0}^{g-1} |\tilde{P}_h|\big\} \tag{26}$$

where $|\tilde{P}_0|$ is set to 0. Their meaning is absolutely analogous to the that for the entire problem, as presented in 3.

Mixed Integer Programming formulation of sub-problems

Once the new sets and parameters are defined, the considered subproblem can be formulated. The formulation uses the same variables presented in Table 1 and inherits Constraints (2)-(8) and (10)-(16), with the only difference being that for each set or parameter, its analogy with the tilde must be used.
Also concerning the objective function, it is totally analogous to (1).

$$\min \sum_{i=0}^{|\tilde{P}|} d_i \tag{27}$$

In terms of constraints, in addition to the inherited constraints, these two must be added:

$$b_g \leq u_k \leq b_g + \varphi_g; \quad g = 1, ..., \tilde{n};\ k = (\tilde{I}_g)_{|\tilde{P}_g|} \text{ only if } |\tilde{P}_g| = |P_g| \tag{28}$$

$$u_k \geq b_g - (|P_g| - |\tilde{P}_g|)\hat{T}_g; \quad g = 1, ..., \tilde{n};\ k = (\tilde{I}_g)_{|\tilde{P}_g|} \tag{29}$$

In particular, Constraint (28) substitutes (9) of the standard formulation. It states that in every SCC g, the last pickup is performed after the closing hour, b_g, within φ_g units of time. Evidently, when dealing with a subproblem, this must be stated only if in the considered subproblem the "real" last pickup of the SCC is included.

Constraint (29) is not included in the original formulation. It is added to ensure that all solutions of the subproblem present partial, feasible solutions for the original subproblem. Therefore, this constraint excludes the feasible region of the subproblem all solutions that are impossible to complete to obtain a full feasible solution. In particular, it is required that, for each center c_g, the last pickup included in the subproblem ($u_{|\tilde{P}_g|}$) is performed not before the closing time b_g is reduced by the maximum time that can elapse between two consecutive pickups at c_g ($\hat{T}_g$) multiplied by the number of pickups excluded from the subproblem ($|P_g| - |\tilde{P}_g|$). Hence, Constraint (29) provides a ***lower limit*** for the pickup visit time k. If this limit is not satisfied, a way to build the next routes is nonexistent without allowing some specimens collected by SCC g to perish.