Publié par : Published by: Publicación de la:

Faculté des sciences de l'administration
2325, rue de la Terrasse
Pavillon Palasis-Prince, Université Laval
Québec (Québec) Canada G1V 0A6
Tél. Ph. Tel. : (418) 656-3644
Télec. Fax : (418) 656-7047

Disponible sur Internet : Available on Internet Disponible por Internet :

http://www4.fsa.ulaval.ca/la-recherche/publications/documents-de-travail/

DOCUMENT DE TRAVAIL 2019-009

An Iterated Local Search for the Biomedical Sample Transportation Problem with Multiple and Interdependent Pickups

Ana Maria ANAYA-ARENAS
Caroline PRODHON
Jacques RENAUD
Angel RUIZ

Document de travail également publié par le Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, sous le numéro CIRRELT-2019-17

Avril 2019

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2019
Bibliothèque et Archives Canada, 2019

ISBN 978-2-89524-488-2 (PDF)

An Iterated Local Search for the Biomedical Sample Transportation Problem with Multiple and Interdependent Pickups

Ana Maria Anaya-Arenas[1,2,*], **Caroline Prodhon**[3], **Jacques Renaud**[1,4], **Angel Ruiz**[1,4]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)
[2] Département de management et technologie, École des sciences de la gestion, Université du Québec à Montréal, 315, rue Sainte-Catherine E., Montréal, Québec, Canada H2X 3X2
[3] Institut Charles Delaunay, Laboratoire d'Optimisation des Systèmes Industriels, Université de Technologie de Troyes, 12, rue Marie Curie, 10004 Troyes, France
[4] Department of Operations and Decision Systems, 2325, rue de la Terrasse, Université Laval, Québec, Canada G1V 0A6

**Corresponding author: anamaria.anayaarenas@cirrelt.ca*

ABSTRACT

This paper addresses a new version of the biomedical sample transportation problem, as a vehicle routing problem with precedence constraints arising in the context of healthcare logistics, and proposes an iterated local search algorithm to solve it. This new version is more realistic and complex since it considers the collection centers' opening hours and the moment at which they are visited as decision variables, granting additional flexibility to elaborate more efficient routes. Indeed, this problem is harder to model and to solve than its previous version because the constraint on the short samples' lifetime leads to interdependency between successive pickups at each collection center. A metaheuristic is thus proposed to solve real-life instances. Numerical experiments confirm (1) the value of simultaneously planning routes, opening hours, and visits' hours (which is new in the literature) and (2) the efficiency of the proposed algorithm to solve this problem.

Keywords: VRP with interdependency, VRP with synchronization constraints, VRP with interdependent time windows, healthcare logistics, biomedical sample transportation, OR in healthcare services, iterated local search.

Acknowledgments: This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), Discovery Grants Program (grants OPG 0293307 and OPG 0172633), the scholarship from the International research internship – Strategic Clusters Program, attributed to Ana Maria Anaya-Arenas, by the Fonds de recherche du Québec - Nature et technologies (FRQNT), and the financial support of Ministry of Healthcare and Social Services of Quebec (MSSS). This financial support is gratefully acknowledged.

1. Introduction

The biomedical sample transportation problem (BSTP), as presented by Anaya-Arenas, Chabot, Renaud, & Ruiz (2016), is a *vehicle routing problem* (VRP) that aims to create a transportation plan to pick up perishable items (biomedical samples) at given locations, referred to as specimen collection centers (SCC), and to take them to facilities possessing the adequate treatment equipment (the laboratories or labs). The centralization of samples' analysis at a single lab brings material advantages from both economic and quality control perspectives. From an economic standpoint, it reduces the investments associated with technical equipment and analysis costs, but it requires a significant logistical effort to transport all the samples to the laboratories. Furthermore, biomedical samples have short lifespans, varying from 5 to 24 hours or more depending on the type of sample. Therefore, SCCs may require several visits per day which leads to a high transportation costs. An efficient transportation plan is thus required to prevent the samples' deterioration and ensuring a high-quality service, while keeping operations' costs as low as possible.

This paper broadens the previous versions of the BSTP according to new requirements expressed by our partner, Quebec's *Ministère de la Santé et des Services sociaux* – MSSS (Ministry of Health and Social Services), which in light of the results and conclusions drawn from our previous works, decided to explore if granting more flexibility to the system's parameters could lead to savings in transportation costs. There are two main differences between this version and the previous one. First, the times at which SCCs are visited to pick up the samples are not considered as defined by the SCCs, but as decision variables that need to be fixed depending on each SCC's service level requirement. Second, we allow some flexibility on SCCs' opening hours seeking a better synchronization of visits and transportation plans. However, this makes the BSTP harder to model and solve because we now need to include constraints to guarantee that a maximum timespan between consecutive pickups is respected. This restriction is expressed through precedence constraints which creates reliance or interdependency between the times SCCs are visited and defines a more complex version of the routing problem, as it will be explained in Section 3. Therefore, the contribution of this article is twofold. Firstly, we define a new and more general version of the BSTP, which is formulated as a VRP with interdependent pickups that also supports tactical decisions for the opening hours of each SCC. As it will be discussed in Section 2, to the best of our knowledge, there is no previous work in the literature that simultaneously addresses (1) the precedence constraints between the multiple visits to the same customer in a single daily routing planning; (2) the route length's limits according to the most urgent item picked up in the route; and (3) the tactical decisions for the visited SCC's opening hours of the to optimize the transportation plan. As a second contribution, we propose an iterated local search (ILS) algorithm to solve the real-life instances of the problem. To efficiently evaluate cost and feasibility whenever a change is introduced in a route, we implement and adapt the concatenation technique of Vidal, Crainic, Gendreau, & Prins (2014) to our problem.

The rest of this paper is organized as follows. Section 2 positions this paper with respect to previous works dealing with (1) synchronization and precedence constraints in routing problems, and (2) works devoted to the transportation of biomedical specimens. Section 3 states the studied version of BSTP and formulates the problem as a MIP, while Section 4 describes the heuristic approach proposed to solve it. Section 5 presents the numerical experiments. Finally, Section 6 draws the main conclusions and research perspectives of our work.

2. Literature Review

The literature on the VRP and its variants is quite extensive and we refer the interested reader to Braekers, Ramaekers, & Van Nieuwenhuyse (2016), Coelho, Renaud, & Laporte (2016) and Lahyani, Khemakhem, & Semet (2015) for recent reviews. Therefore, we concentrate our efforts in reviewing contributions that share at least one of the two characteristics that make our problem more than a straightforward VRP: (1) the interdependency between successive visits to customers (SCC), and (2) the lifetime of the products (samples) that limits the time available to return to the depot (the laboratory).

Time constraints in customers' service have traditionally been addressed through the definition of time windows –TW (see, for example Bräysy & Gendreau 2005a,b). Some practical contexts have raised a need to define multiple (but independent) time windows for each customer to perform one or several visits (e.g., Amorim, Parragh, Sperandio, & Almada-Lobo, 2014; Belhaiza, Hansen, & Laporte, 2014; Favaretto, Moretti, & Pellegrini, 2007; Tricoire, Romauch, Doerner, & Hartl, 2010). The pickup and delivery problem (PD) defines a particular class of dependency between visits, which requires that a given pickup point must be scheduled before its associated delivery point. For instance, Xu, Li, Zou, & Liu (2017) proposed a PDVRP that allows (but does not require) multiple visits to the same customer. Dumas, Desrosiers, & Soumis (1991) were among the first to consider time windows in a pickup and delivery VRP, which is referred to as PDVRPTW. Since then, a number of works have formulated healthcare settings, and particularly homecare contexts, as PDVRPTW. For instance, Liu, Xie, Augusto, & Rodríguez (2013) proposed a PDVRPTW formulation to model home care planning where a single visit to a patient's home can be used to simultaneously deliver (from a hospital and/or a depot) and pick up goods at patients' homes that have to be taken later to the depot or the lab. However, Liu et al. (2013) did not consider any restriction on samples' transportation, as they were taken to the lab at the end of the route. Lim, Zhang, & Qin (2016) proposed a multi-trip PDVRPTW that manpower scheduling for a non-emergency ambulance transportation system.

Fewer contributions consider temporal precedence and/or synchronization constraints between visits or activities. However, this feature is often found in practice, gaining more attention from the operations research community in the last decade (see Drexl, 2012, for a review). Ioachim, Desrosiers, Soumis, & Bélanger (1999) were among the first to study this time restriction in a fleet assignment problem for aircraft routing. Later, Bredström & Rönnqvist (2008) described some contexts that require this type of constraints and proposed a general model using synchronization constraints explicitly. These constraints were later generalized by Dohn, Rasmussen, & Larsen (2011). Rousseau, Gendreau, & Pesant (2013) presented the synchronized dynamic vehicle dispatching problem and El Hachemi, Gendreau, & Rousseau (2013) solved a synchronized VRP emerging from the forestry industry where some customers require a simultaneous visit of two types of vehicles. Goel & Meisel (2013) presented a routing and scheduling problem for jobs maintenance with precedence constraints, however, in this case there was no time limit or maximum length to plan the schedule.

Other dependencies between visits arise when visits are performed with a given frequency on the planning horizon (usually a week) as it is the case, for instance, in the service consistency requirements for the inventory routing problem (e.g., Coelho, Cordeau, & Laporte, 2012) or the routing planning for the waste collection problem (e.g., Huang & Lin, 2015). Applications like tramp ship routing and scheduling problem also require the interdependency between the visits, but with no routing restrictions (e.g. Andersson, Duesund, & Fagerholt, 2011; Bakkehaug, Rakke, Fagerholt, & Laporte, 2016; Norstad, Fagerholt, Hvattum, Arnulf, & Bjørkli, 2015; Vilhelmsen, Lusby, & Larsen, 2014).

Home care planning, where a visit to a patient must be planned within a specific time window and not after (or before) a certain activity (or a previous visit), is a practical context encompassing visit interdependency. Fikar & Hirsch (2015) planned a multimodal problem with synchronization constraints, time windows and break scheduling. Rasmussen, Justesen, Dohn, & Larsen (2012) included the preferences of patients in the assignment of caregivers. Mankowska, Meisel, & Bierwirth (2014) presented a similar problem, and also included the possible heterogeneity of the staff's vehicles. Finally, Labadie, Prins, & Yang (2014) studied the synchronization constraints and their model was later extended by Haddadene, Labadie, & Prodhon (2016), which also proposed a greedy randomized adaptive search procedure (GRASP) and an ILS embedded metaheuristic to solve it.

The second characteristic of our problem is the route duration limit which, unlike most VRPs, is related to the most urgent client (collected sample) in a route. This kind of constraint is more common in other applications like passengers' transportation. Indeed, our problem can be viewed as an extension of the dial-a-ride problem – DARP (see Cordeau & Laporte, 2007 for a review). Like the DARP, our problem has precedence and maximum transportation time constraints. However, the multiple requests to a single customer (with the interdependence) make BSTP more complex. To the best of our knowledge, Reinhardt, Clausen, & Pisinger (2013) **is** the only contribution with synchronization constraints for the DARP. Reinhardt, Clausen, & Pisinger (2013) studied a multi-modal optimization model for planning the transportation of passengers with reduced mobility in an airport, synchronizing different agents, buses and shuttles. However, they imposed specifics TW for each pickup and sought to minimize the transfer's total duration, while this time is restricted in our case.

The maximum time constraint to return to the depot is common in blood or biomedical sample management problems due to the short lifespan of specimens (see Baş, Carello, Lanzarone, Ocak, & Yalçindag, 2016). Pioneer works on blood logistics problems appeared in the 1970s (McDonald, 1972), but they have regained attention in recent years. Yi (2003) was among the first to approach the blood transportation problem as a variant of the VRPTW, where only one pickup has to be performed at each center and the objective is to maximize the quantity of blood that is collected and treated, while minimizing transportation costs. On their side, Sahinyazan, Yetis-Kara, & Taner (2015) presented a tour mobile collection system for the Red Cross in Turkey. Yücel, Salman, Gel, Örmeci, & Gel (2013) included the laboratory's processing rate in the tour's planning to seek balance between the number of samples processed in a day and the transportation costs. Contrarily to the BSTP, these routes are not restricted by any time constraints. Ghandforoush & Sen (2010) proposed a decision support system to manage platelet production that only allowed round trips for transportation. Mobasher, Ekici, & Özener (2015) proposed a similar problem including appointment planning at the collection centers. These last two studies considered a maximal arrival time at the lab, but just a single pickup at each location, so there is no interdependency in route planning. Closely related, Anaya-Arenas et al. (2016) presented the BSTP as a multi-trip VRPTW, where each collection center requires several visits within independents time windows and the samples' lifespans limit the duration of the routes. Naji-Azimi, Salari, Renaud, & Ruiz, (2016) presented a variant where they minimized the number of trucks' arrivals to the lab in each time slot. However, they still considered independent time windows for each pickup. Kergosien, Ruiz, & Soriano (2014) elaborated routes for nurses who visit patients at their homes and several drop-off points were scattered over the region, allowing nurses to deposit their samples to respect samples' lifespan. Even if some of these contributions considered a route length limit, their pickup decisions did not include any interdependency.

To the best of our knowledge, the only works considering interdependency in the pickup decisions and the routes' length limitation, as in our context, are Doerner, Gronalt, Hartl, Kiechle, & Reimann (2008) and Doerner & Hartl (2008). They presented the Austrian Red Cross' blood collection process, which relies on mobile collection sites that cannot ensure the conditions to preserve samples' quality. Consequently, several pickups must be planned for every collection point, and the time of each pickup depends on when the previous one was performed. The authors proposed a complex MIP formulation to model the problem and, since solving even small sized instances required days of computing, a fast construction heuristic. Even if their context is similar to ours, in their case, each SCCs' operation hours were given in advance, thereby reducing the complexity of the timing decisions. Therefore, we can conclude that none of the previous works attempt to simultaneously define (1) the moments at which pickups are performed (2) the SCC's opening hours, and (3) restrict the transportation duration as we need to do it. Since these decisions are linked, considering them altogether within an optimization approach may result in interesting savings. However, doing so introduces very challenging difficulties, both at the formulation and solving steps, as it will be explained in the following sections.

3. Problem Statement and Formulation

This section presents the characteristics of the BSTP studied in this paper. First, we describe the problem in a formal manner and illustrate its difficulty in section 3.1. Then, the mathematical formulation is presented in section 3.2.

3.1. The BSTP with Interdependent Pickups

The biomedical sample collection network in the Province of Quebec is divided into a number of independent administrative clusters, each being composed of a set of SCCs and a laboratory. In the following, operations in each cluster are described and modelled. We define $N = \{c_1, c_2, \dots, c_n\}$ as the set of the n sample collection centres, SCCs, assigned to the lab. Each SCC c_g has a specific timespan (named collection period O_g), during which samples are taken, pretreated and placed in standard coolers for transportation to the lab. Each SCC c_g has some flexibility for the starting time of the collection period (named a_g), which needs to be set between e_g and l_g (the earliest and latest time at which SCC c_g can open to the patients). Thus, once a_g has been selected, the SCC's end time for its collection period is $b_g = a_g + O_g$.

Moreover, in an attempt to ensure a high level of service at the SCCs and to maintain a certain stability in the samples' arrival at the laboratory, each SCC c_g imposes a maximum time Δ_{max}^{g} between two consecutive pickups. Δ_{max}^{g} is negotiated according to the level of service desired by the SCC or by the Ministry. Consequently, SCC c_g requires a minimum number of visits, which will be referred to as *transportation requests*, *pickups* or simply *visits* (indistinctly) in the sequel. We define P_g as the set of visits requested by SCC c_g, and $|P_g| = \lceil O_g/\Delta_{max}^{g} \rceil$. It is worth mentioning that, in all the cases, the last visit to each SCC must be scheduled after the end of the collection period (after b_g) to guarantee that all the samples are taken to the lab. We define φ_g as the maximal additional time allowed by the SCC c_g (after b_g) to perform the last visit of the day. This maximal time can be defined by the working hours of the SCC's staff or their service level's requirements. As soon as the samples are out of the SCC's controlled environment, they must arrive to the lab within a transportation time limit related to the biological degradation of the samples, their nature and the type of tests to perform. It is thus possible to define a *maximal transportation time* for a given transportation request p (T_{max}^{p}) as the shortest transportation time limit among all the samples picked up at p.

These two particularities make this problem a very difficult one. The maximal time between consecutive pickups and the opening flexibility creates interdependency between the visits. In addition, the maximal transportation time impacts the length of the routes and interrelate them. To illustrate these difficulties, let us consider an example with a SCC c_g, and a collection period of O_g as shown on Figure 1. To simplify the example, we assume that four visits need to be performed at SCC c_g. We define u_i (with $i = 1,2,3,4$) as the pickup time of visit i, and of course, as a general rule $u_i < u_{i+1}$.

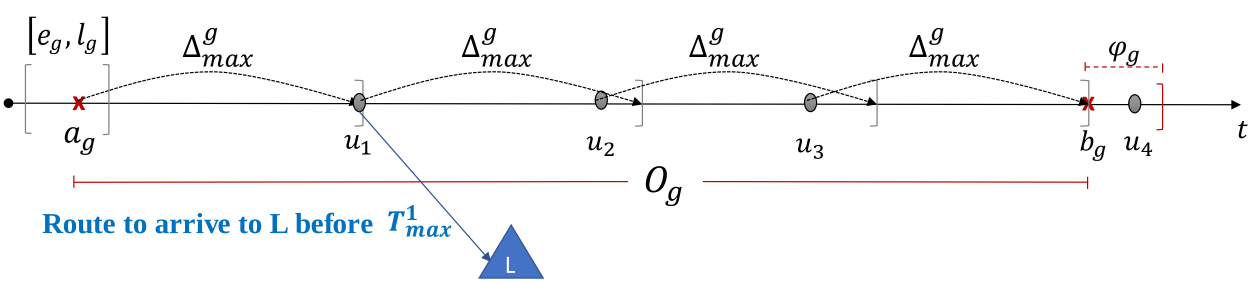

Figure 1 – Example of interdependency between pickups at SCC c_g.

The SCC c_g opens at instant a_g and thus closes at time $b_g = O_g + a_g$. To satisfy the required level of service, a first collection visit must be performed in the interval $(a_g; a_g + \Delta_{max}^{g}]$. Let us assume, for instance, that this first visit happens at time u_1, with $a_g \leq u_1 \leq a_g + \Delta_{max}^{g}$. Then, it follows that a second collection (at time u_2) must satisfy $u_2 \leq u_1 + \Delta_{max}^{g}$ and, in general, $u_i \leq u_{i-1} + \Delta_{max}^{g}$, until the last pickup visit is scheduled after the closing time and before the limit φ_g (i.e. $b_g \leq u_{|P_g|} \leq b_g + \varphi_g$). Notice that a visit can be performed at any time before the Δ_{max}^{g} limit (as it is the case of u_2 and u_3), but performing an earlier pickup may force the next visits to be performed earlier as well. Let us mention that the flexibility in the SCCs' opening hours brings an additional level of complexity to this problem: the decision on when to perform the first visit (u_1) is related to the decision of opening time (a_g). Also, the closing time must respect the Δ_{max}^{g}

with the previous pickup time (u_3) and the maximal time to perform the last pickup after closing. Finally, remember that once a pickup i has been set in a route (at u_i), the samples must arrive at the lab before T_{max}^i time units to preserve their quality. Therefore, the interdependency also implies a strong relationship with the routing decisions.

3.2. MIP for a Fixed Number of Pickups

Based on the results of Anaya-Arenas et al. (2016), where the extended graph formulation proves to be more efficient, we model the BSTP with interdependent pickups over a complete graph $G = \{V, A\}$, where the set of nodes $V = \{v_0, v_1, v_2, \dots, v_{|P|}\}$ is formed by the $|P|$ transportation requests of all SCCs ($P = \cup_g P_g$), and the laboratory ($\{v_0\}$) where the routes must start and end. Without loss of generality, we label the pickups so that $\{v_1, v_2, \dots v_{|P_1|}\}$ are the transportation requests of SCC c_1, $\{v_{|P_1|+1}, \dots, v_{|P_1|+|P_2|}\}$ are requests of SCC c_2 and so on. More precisely, we define I_g as the index set for the requests of a center c_g where $I_g = \{\sum_{h=1}^{g-1}|P_h| + 1, \dots, \sum_{h=1}^{g}|P_h|\}$. In addition, we define an arc set $A = \{(v_i, v_j): v_i, v_j \in V, i \neq j, i, j = 0, \dots, |P|\}$ and for each arc (v_i, v_j) a fixed transportation time (t_{ij}) is known. Lastly, $t_{ij} = 0$ if v_i and v_j belong to the same SCC ($v_i, v_j \in P_g$).

The objective is to elaborate the transportation plan with the minimum total duration, defining the SCCs' opening hours and a set of routes visiting the SCCs in such a way that all the service's constraints are respected and none of the samples perish. In addition to the notation already defined, we set T_{max} to control the maximal transportation times of the samples ($T_{max} = \min_i T_{max}^i$), we define and δ_i as the difference (if any) between T_{max}^i and T_{max} ($\delta_i = T_{max} - T_{max}^i$). The routing time must also account for a loading time τ_g at SCC c_g and the unloading time of the vehicle at the lab (τ_0) before a new route can begin. Lastly, let us introduce the following decision variables:

x_{ij} takes value 1 if node i is visited before node j,
u_i indicates the time at which pickup i is performed,
d_i calculates route's duration in which pickup i is the first pickup performed,
f_i indicates the time remaining at node i to complete the route and to bring the most urgent request to the lab, still respecting the T_{max}^i,.
a_g indicates the start time of the collection period of SCC c_g.
b_g indicates the finishing time of collection period at SCC c_g.

The BSTP can thus be modelled as a MIP as follows:

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

Subject to:

$$\sum_{i=0}^{|P|} x_{ij} - \sum_{i=0}^{|P|} x_{ji} = 0 \; ; j = 0, \dots, |P| \quad (2)$$

$$\sum_{i=0}^{|P|} x_{ij} = 1 \; ; j = 1, \dots, |P| \quad (3)$$

$$u_j \geq u_i + \tau_i + t_{ij} - M(1 - x_{ij}) \; ; i = 0, \ldots, |P|; \; j = 1, \ldots |P|; \; (i \neq j) \tag{4}$$

$$e_g \leq a_g \leq l_g \; ; g = 1, \ldots, n \tag{5}$$

$$a_g + O_g = b_g \; ; g = 1, \ldots, n \tag{6}$$

$$u_k - a_g \leq \Delta_{max}^g \; ; g = 1, \ldots, n \text{ where } |P_g| > 1; \, k = \sum_{h=1}^{g-1} |P_h| + 1 \tag{7}$$

$$u_k - u_{k-1} \leq \Delta_{max}^g \; ; g = 1, \ldots, n \text{ where } |P_g| > 2; \, k = \sum_{h=1}^{g-1} |P_h| + 2, \ldots, \sum_{h=1}^{g} |P_h| - 1 \tag{8}$$

$$b_g - u_k \leq \Delta_{max}^g \; ; g = 1, \ldots, n \text{ where} |P_g| > 1; \, k = \sum_{h=1}^{g} |P_h| - 1 \tag{9}$$

$$b_g \leq u_k \leq b_g + \varphi_g \; ; g = 1, \ldots n; k = \sum_{h=1}^{g} |P_h| \tag{10}$$

$$T_{max} - f_i + M(1 - x_{i0}) \geq t_{i0} + \tau_i \; ; i = 1, \ldots, |P| \tag{11}$$

$$f_j - f_i + M(1 - x_{ij}) \geq u_j - u_i \; ; i, j = 1, \ldots, |P| \; (i \neq j) \tag{12}$$

$$f_i \geq \delta_i \; ; i = 1, \ldots, |P| \tag{13}$$

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

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

$$x_{ij} = \{0,1\} \; ; i \neq j = 0, \ldots, |P| \tag{16}$$

The objective function (1) minimizes the total routes' duration. It will be explained hereafter with the example of Figure 2. Constraints (2) ensure flow conservation on every node of the graph, while constraints (3) ensure that every visit j is satisfied. Constraints (4) to (10) are time constraints. Constraints (4) estimate the collection time for visit j and eliminate the sub-tours between the nodes. Constraints (5) ensure that SCC c_g starts its collection period inside its given time window and constraints (6) set the end time of the collection period. Constraints (7) ensure that the first visit at SCC c_g (visit $k = \sum_{h=1}^{g-1} |P_h| + 1$) is performed before Δ_{max}^g units of time after the SCC c_g starts collecting samples. Constraints (8) verify that all pairs of consecutive visits at SCC c_g (k and $k-1$, with $|P_g| > 2$ and $k = \sum_{h=1}^{g-1} |P_h| + 2, \ldots, \sum_{h=1}^{g} |P_h| - 1$) satisfy the maximal timespan Δ_{max}^g. Constraints (9) ensure that the SCC c_g's collecting period ends no later than Δ_{max}^g minutes after the penultimate visit. Clearly, constraints (7) and (9) are imposed exclusively over the SCCs demanding more than a single visit during its collection period, and constraints (8) are only needed if the SCC c_g requests three visits or more. In addition, constraints (10) state that the last visit to SCC c_g ($k = \sum_{h=1}^{g} |P_h|$) is performed after b_g, but before the φ_g time limit. Constraints (11) to (13) control the flow of time restrictions over all the visits. For any visit i, we need to fix a time limit to return to the lab from the moment the visit is done, rather than from the moment the vehicle starts a route. We therefore use a "remaining time" resource variable. To compute the remaining time, we calculate the time that is consumed from the lab to each customer in the opposite direction of the route. This time consumption is directly related to the time at which each visit is done. To better illustrate the way that these constraints works, we use a numerical example of a single route, which is illustrated in Figure 2. The travel time is presented near each arc and, to simplify, we assume that the loading and unloading times are zero, and that there is no waiting time at any node.

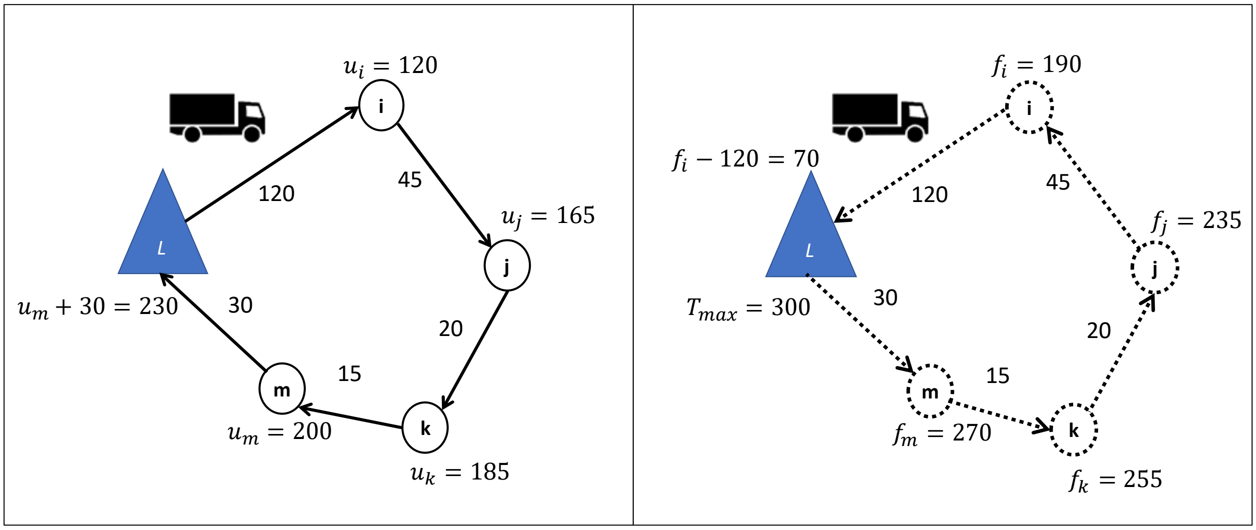

Figure 2 – Routing time calculation (left) and remaining time consumption (right)

On the left side of the Figure 2, the route duration is illustrated with the arrival time at each node (variable u). Supposing that the truck leaves the lab (L) at time 0, it arrives to perform the pickup i at instant 120 ($u_i = 120$). As the loading time is zero, we have $u_j = 165$, $u_k = 185$, $u_m = 200$ and the route finishes at 230. On the right side of Figure 2, the time consumption is illustrated considering that T_{max} is set to 300 minutes. The remaining time consumption is computed in the opposite direction of the route. Therefore, the remaining time resource at node m is 270 minutes ($f_m = 270$), $f_k = 255$, $f_j = 235$, $f_i = 190$ and, finally, the remaining time at the beginning of the route is 70 minutes ($f_i - 120 = 70$). This route duration is 230 minutes, and it can be computed either by the sum of the transportation time (on the left side of Figure 2) or by using constraint (14) which gives $T_{max} - f_i + t_{0i} = 300 - 190 + 120 = 230$ (on the right side of Figure 2). This behaviour is generalized by constraints (11)-(14) as follows. Constraints (11) ensure the coherence between the time resource variables and the transportation and loading time for any node i, if the node i is visited right before returning to the depot (the last node of any route). Constraints (12) do the same for any pair of nodes (i, j), saying that if the arc (i, j) is included in the route, the difference between its respective service time variables must match the resource consumption difference. Constraints (13) force the time resource left at node i to be greater than δ_i in order to respect the maximum transportation time of pickup i. Finally, constraints (14) estimates the duration of routes, stating that if the route first visits the node i, its duration is greater or equal to the maximum route duration Tmax, minus the time consumed from the Lab to i, plus the travel time of arc $(0, i)$ and the unloading time at the depot. If node i is not the first node of the route (i.e. $x_{0i} = 0$), the duration will be fixed to 0. Constraints (15) and (16) define the decision variables' domain.

This formulation, implemented in Gurobi (v.6.5.1), allows us to solve small sized instances to optimality in short computational times and to identify a feasible solution for all instances in the first seconds. However, the computational time required quickly grows with the problem's size, and it becomes impossible to reach an optimal solution, in a reasonable time, for the large size, real-life inspired instances. We were hence motivated to develop the approximated method described in the following section.

4. An Iterated Local Search for the BSTP

This section proposes an iterated local search (ILS) procedure to efficiently solve the real-life instances of the BSTP. The method encompasses one initialization and two-nested diversification stages. In the *Initialization* stage, a feasible initial solution S is produced by a construction heuristic and a local search procedure (LS) is applied to S until no further improvement can be reached. Then, the algorithm iterates between stages 2 and 3 for a given number of times it_1. In stage 2, a "soft" adaptive diversification is performed on the current solution. Basically, this diversification removes a certain number of SCC from the solution following a given strategy, re-optimize the remaining routes using the local search procedure, and then reintroduces the removed SCCs back into the solution. Finally, the local search procedure is applied to this new solution. The number of SCC to remove, as well as the way in which they are chosen, are given by parameters that are adjusted according to the search status, as it will be described in Section 4.2. Stage 2 is repeated for a pre-defined number of times it_2 or until all of the adaptive search's options have been applied without improving the best solution found so far. Stage 3 performs a hard diversification on the current solution. In short, following a given strategy, all the SCC are removed from the current solution and reinserted to produce a new feasible solution. As it will be explained later, the hard diversification is not a simply restart. On the contrary, the remove-and-insert strategy considers the search history to progressively lead the search towards different regions of the solution space. After executing Stage 3, the algorithm moves back to Stage 2 or, if the iterations limit it_1 is reached, the search is stopped. A pseudocode of the proposed ILS is given in Figure 3. The three stages of the algorithm are described in the following subsections.

Initialization

```
Built initial solution
Local search
Repeat it_1 times
    Repeat it_2 times
        Remove ϑ SCCs from S
        Local search
        Insert removed SCCs back into S
        Local search
        Update search parameters
    End Repeat
    Remove N − 1 SCCs from S
    Insert removed SCCs back into S
    Local search
End Repeat
```

Soft diversification

Hard diversification

Figure 3. Pseudocode of the ILS algorithm

4.1. Initialization

A constructive heuristic is used to build an initial solution as follows:

1. Create a list P containing all the transportation requests (visits).
2. For each visit v in P, we compute the initial time bounds and concatenation parameters (as explained in Appendix 1 and 2).
3. A first route r is created with the first visit v in P, and it is added to the solution S. Parameters of visit v and all the visits in P are updated. v is removed from P.
4. The next visit in the list (v') is inserted in the best possible location in the existing routes or in a new route (r') of the solution S. Time bounds and concatenation parameters are updated for all visits in s and in P, and v' is removed from P.
5. Repeat step 4 until all the visits are scheduled in the solution S.

The variable *best*, which records the best solution found so far, is set to S. The feasible solution S is then passed to the local search procedure which is described now.

Local Search Procedure (LS). Local search is based on the exploration of solutions that are "near" (i.e. within a fixed distance) the current solution. Neighbourhood operators are mechanisms defining how the neighbour solutions are set. In order to restrict the neighbourhood's size, and therefore the computational effort to explore them, we propose the use of two operators that evaluate changes in the solution that concern only one transportation request at a time. The first operator, **N1**, attempts to relocate a given transportation request in a different position in its route. The second operator, **N2**, evaluates potential relocations of a given transportation request in other routes. These two operators offer complementary effects and different advantages and drawbacks, so they need to be used at different moments of the search and with carefully chosen frequencies. To this end, we have embedded them in a variable neighbourhood descent (VND) scheme, which starts with **N1** and changes to **N2** when no further improvement is possible. If **N2** leads to a better solution, it is implemented and the algorithm returns to **N1**. However, if **N2** does not achieve an improvement, a local optimum is found and thus the local search is finished. Each time the local search procedure finds a solution S' which improves the best solution found so far, *best*, then *best* is updated (*best* ← S').

In the local search we elected to explore the neighbourhoods in an exhaustive manner selecting the move which gives the best improvement in the objective function value. In the case of **N1**, this means that, starting with the first route and the first transportation request, we explore all the possible relocations of every transportation request in the route, and we then proceed similarly with each of the remaining routes. When all the relocations have been evaluated, the one leading to the best improvement is implemented, and a new iteration using **N1** is launched. It is worth mentioning that many moves can be unfeasible, due to the strong interdependency between the visits and the time constraints. The bounds explained in Appendix 1, and the concatenation techniques detailed in Appendix 2, allow us to quickly identify such cases, thus improving the VND's efficiency.

4.2. Soft Diversification

The *Soft diversification* procedure aims at modifying the structure of the current solution in an adaptive manner to escape from the local optimum reached by the LS procedure. To this end, it proceeds as follows. Based on a given strategy (see later), all the visits of one or several SCCs are removed from S to create a partial solution $\check{S}$. Then, the LS is applied to $\check{S}$ to reorganize the routes. Finally, the visits associated with the removed SCCs are reinserted in the best possible positions, leading to a new complete solution S'. The LS procedure is called again to be applied on S', and then a new S' solution is produced. Notice that nothing ensures that the new S' improves $best$. Therefore, we evaluate the quality of the solution at the end of each iteration. If S' is better than *best*, then *best* is updated. Otherwise, we keep S', but a counter named $failed$ is increased.

To ensure that the procedure achieves the desired diversification and that the algorithm does not cycle between already explored solutions, we developed two kinds of “tabu-like” memories. The first one forbids the insertion of the removed SCC’s in S' in the same route as it was in S. The second one prevents a SCC that was been removed and reinserted during the last *failed* iterations from being considered for the diversification.

As per the diversification strategy, there is no simple way to decide which SCC should be removed, nor is there an easy way to anticipate its impact on the whole solution. Therefore, at each execution, the algorithm randomly chooses one of the following selection rules:

1) Highest detour (HD): select the SCC with the visit that produces the highest increase in the route’s length duration.
2) Mean highest detour (MHD): select the SCC that has the highest average detour among all its visits.
3) Random (Rnd): select randomly the SCC to remove.

At the beginning of the algorithm, each rule has the same probability π_i, $i = \{1, 2, 3\}$ to be selected ($\pi_i = 1/3$), and then probabilities are adjusted at the end of each iteration of the *Soft diversification* step in the following manner: if the selection rule that was applied led to an improvement of *best*, its probability is increased by γ and the probability of the other two rules is decreased by $\gamma/2$ each. Otherwise, its probability is decreased by γ and the probability of the other two rules is increased by $\gamma/2$. Nonetheless, the probabilities are limited to the $\{0.15; 0.70\}$ interval in all the cases.

Also, the *Soft diversification* step includes an adaptive strategy that dynamically doses the strength of the diversification procedure according to the status of the search by changing ϑ, the number of SCCs that are removed from the incumbent solution. At the beginning of the algorithm, we set $\vartheta = 1$ and the search is performed on the region of the solution space around S, which is hopefully improved. When the $failed$ counter reaches the $failed\_max$ limit, it is necessary to move the search to other farther regions. To do so, ϑ is increased gradually until it reaches a maximum value ϑ_max. After ϑ_max SCCs have been removed, the last mechanism used to enhance the robustness of the algorithm is activated, forcing the diversification of the SCCs’ opening hours. After an SCC is removed during the procedure, its opening time when reinserted in S' will be shifted as far as possible (earlier or later) within its opening time window. If after this the algorithm is still unable to find a new best, the *Soft diversification* stage is stopped and the algorithm then moves to the *Hard diversification* stage.

4.3. Hard Diversification

The *Hard diversification* aims to move the search to new regions of the solution space once a given region has been explored. To this end, all but one of the SCCs are removed from solution *best*, and reinserted in the best possible manner. SCCs are removed according to the three selection rules described in the previous subsection and their current probabilities, so the new solution produced by the procedure depends on the search history. In fact, the vector π_i which contains the probabilities for each select rule can be seen as a long-term memory for the algorithm. Finally, notice that the algorithm will execute up to it_1 times the *Hard diversification* step.

5. Numerical Results

In this section, we present the computational experiments executed (1) to evaluate the extent to which granting some flexibility to the SCCs opening hours may lead to reductions on the routes duration, and (2) to analyze the performance of the ILS algorithm.

The algorithm was coded on VB.NET and executed on a PC with an Intel® Core™ i7-6700 CPU @3.4 GHz (2 processors), 16 GB of RAM. We solved the set of 38 instances proposed in Anaya-Arenas et al. (2016). Instances are separated in two groups. The *small and medium* groups have 27 instances, with up to twelve SCCs and a maximum of 21 transportation requests to schedule. The *large* group includes 11 instances, with up to 20 SCCs and a maximum of 50 transportation requests to schedule. We refer the interested reader to Anaya-Arenas et al. (2016) for a thorough description of the instances and the way in which they were built.

5.1. Effect of the Flexibility in the Opening Hours of SCCs

Although SCCs remain open for fixed collection periods, a small amount of flexibility on their opening hours may be acceptable. Indeed, we hypotetize that, by doing so, collection routes might be improved because SCCs requests would tend to be desynchronized (i.e. SCCs will not require to be visited at the same time). After discussion with managers from the MSSS, we agreed on a flexibility period of 15 minutes around the SCCs current opening hours. It is worth mentioning that a larger opening time window could lead to better solutions, but managers considered that modifying employees' habits might be too complicated.

To assess the impact of such flexibility on the quality of the solutions, we solved two different versions of the mathematical model proposed in section 3.2. In the first version (named *Rigid*) no flexibility was allowed and we set the opening hours of the SCCs to the times provided by the MSSS (i.e. $e_g = l_g = a_g$). In the second one, called *Flexible*, and according to the MSSS's suggestion, we allowed opening times within a window of 15 minutes before and after its current opening hours. Both models were solved using Gurobi (V.6.5.1) with a computational time limit of 36 000 seconds (ten hours). Results are presented in Table 1, where the three leftmost columns define the instances in terms of the number of SCCs (Column $|N|$) and the number of transportation visits to schedule (Column $|P|$). Then, columns four to six reports the results produced by the fixed opening hours' model (*RG*), while columns seven to nine reports the results produced by the formulation which include flexibility in the opening hours of the collection periods (*FL*). In both cases, columns *Obj.* , Sec., and Gap% report the value of the objective function, the computational time and the optimality gap reported by the solver, respectively. Notice that the solver estimates the optimality gap as Gap%= (Upper bound – Lower Bound) / Upper Bound. Finally, we present the difference in percentage of the objective function obtained when we allow the time windows for the opening hours in the formulation. This value is given in column 10 (%) and it is calculated as $\% = (FL\ Obj. - RG\ Obj\ )/RG\ Obj.$

Table 1 shows that, for small and medium-sized instances (I01 to I27), Gurobi can solve the rigid (RG) model and flexible (FL) model to optimality in a short computational time. Moreover, we can see that including flexibility in the SCCs' opening hours led to better solutions (shorter routing duration) in 19 out of 27 instances, with an average and maximal reductions on the total collecting time of 4.89% and 16.59%, respectively. We can thus conclude that in the first 27 instances tested, a small amount of flexibility in the SCCs' opening hours leads to a better synchronization of visits and therefore better transportation plans. This result justifies, in our opinion, the development of a model and solutions approaches to tackle opening time decisions efficiently.

Table 1 – Rigid vs. flexible opening hours versions

Instance			***RG* (36000 s.)**			***FL* (36000 s.)**			**%**
	|N|	**|P|**	Obj.	Sec.	Gap%	Obj.	Sec.	Gap%	
I01	2	3	270	0.0	0.00%	270	0.0	0.00%	0.00%
I02	3	3	179	0.2	0.00%	174	0.1	0.00%	**-2.79%**
I03	2	4	383	0.0	0.00%	358	0.4	0.00%	**-6.53%**
I04	3	4	308	0.2	0.00%	308	0.0	0.00%	0.00%
I05	3	6	444	0.1	0.00%	404	0.4	0.00%	**-9.01%**
I06	5	7	391	0.4	0.00%	391	0.3	0.00%	0.00%
I07	5	7	383	0.4	0.00%	383	0.4	0.00%	0.00%
I08	5	7	384	0.4	0.00%	384	0.5	0.00%	0.00%
I09	4	7	422	0.2	0.00%	352	0.5	0.00%	**-16.59%**
I10	2	7	628	0.2	0.00%	628	0.3	0.00%	0.00%
I11	4	7	716	0.4	0.00%	716	0.2	0.00%	0.00%
I12	4	7	318	0.4	0.00%	305	0.4	0.00%	**-4.09%**
I13	4	8	853	0.2	0.00%	815	0.4	0.00%	**-4.45%**
I14	5	8	349	0.4	0.00%	347	0.6	0.00%	**-0.57%**
I15	5	8	515	0.4	0.00%	461	0.2	0.00%	**-10.49%**
I16	4	9	304	0.5	0.00%	304	0.5	0.00%	0.00%
I17	4	9	345	0.5	0.00%	313	0.5	0.00%	**-9.28%**
I18	4	9	347	0.4	0.00%	320	0.5	0.00%	**-7.78%**
I19	6	9	599	0.2	0.00%	512	0.4	0.00%	**-14.52%**
I20	6	11	501	0.6	0.00%	485	2.9	0.00%	**-3.19%**
I21	9	13	460	23.3	0.00%	412	37.2	0.00%	**-10.43%**
I22	8	13	560	1.5	0.00%	497	9.9	0.00%	**-11.25%**
I23	7	14	1 075	1.1	0.00%	1 058	3.8	0.00%	**-1.58%**
I24	9	16	1 195	1.5	0.00%	1 185	16.4	0.00%	**-0.84%**
I25	9	16	1 247	0.3	0.00%	1 167	2.0	0.00%	**-6.42%**
I26	12	18	1 432	19.4	0.00%	1 343	859.4	0.00%	**-6.22%**
I27	11	21	2 052	1.2	0.00%	1 928	726.8	0.00%	**-6.04%**
Average			**617**	**2.0**	**0.00%**	**586**	**61.7**	**0.00%**	**-4.89%**
I28	17	24	2 602	104.5	0.00%	2 088	36 000	47.73%	**-19.75%**

I29	10	30	1 014	36 000	93.81%	951	36 000	100.00%	-6.17%
I30	11	32	1 053	36 000	100.00%	990	36 000	100.00%	-6.03%
I31	12	34	1 147	36 000	100.00%	1 089	36 000	100.00%	-5.10%
I32	17	35	1 867	36 000	65.23%	1 786	36 000	76.08%	-4.34%
I33	13	36	1 177	36 000	100.00%	1 147	36 000	100.00%	-2.55%
I34	18	39	2 034	36 000	79.65%	1 881	36 000	86.81%	-7.52%
I35	19	40	2 143	36 000	73.92%	1 985	36 000	76.64%	-7.37%
I36	19	40	2 209	36 000	74.83%	2 039	36 000	79.10%	-7.70%
I37	19	41	2 319	36 000	76.39%	2 112	36 000	81.16%	-8.93%
I38	17	50	1 199	36 000	100.00%	1 170	36 000	100.00%	-2.42%
	Average		**1 706**	**32 736.8**	**78.53%**	**1 567**	**36 000.0**	**86.14%**	**-7.08%**

If we look at the large-sized instances, the solver was unable to provide proof of optimality within the allotted time (36 000 seconds), except for instance I28, for which the rigid optimal solution is improved by the flexible one by at least 19.75%. Finally, as I29 to I38 could not be solved to optimality inside the computational time limit using the RG or the FL formulation, we cannot fairly compare their performance. Nonetheless, it is possible to see that, inside the same computational time limit, the FL model achieved a lower value of the objective function than the RG model.

5.2. Algorithm efficiency

This section seeks to assess the ability of the ILS algorithm to produce good quality solutions for formulations (1) to (16), particularly for the large-sized instances that could not be solved efficiently by the commercial solver. Before getting to the produced results, let us explain that a set of preliminary experiments led us to set the ILS's parameters to the following values:

- Limit it_1 = 5× $|N|$;
- Limit it_2 = 2 × $|P|$;
- Number of maximum failed iterations failed_max = 2;
- Maximum number of SCCs to remove from the solution at the diversification procedure, $\vartheta\_\max = \min(4\ ;\ 0.1|N| + 1)$;
- SCC selection rule adjustment parameter, $\gamma = 0.1$

To make the results easier to read, they were separated according to the size of the instances. Hence, Table 2 presents the results for the small and medium instances, while Table 3 does it for the larger instances. In both cases, the three leftmost columns define the instances in terms of the number of SCCs (Column $|N|$) and the amount of transportation visits to schedule (Column $|P|$). Then, columns four to six show the results produced by *Gurobi* in terms of the best objective function (Column *Obj*.), the computational time (Column *Sec.*) and the relative MIP optimality gap (*Gap%*) as reported by *Gurobi*. Column seven presents the results produced by the *initialization stage* of the algorithm (i.e. the construction heuristic and then a local search) in terms of its relative distance to Gurobi's best known solution (%*GUR*). Columns 8 to 11 present the results of the ILS algorithm. Since there are random components on the ILS, the reported solution can vary from one execution of the algorithm to the other. We therefore performed ten independent executions (runs) of the algorithm and reported the total execution time for the 10 runs (column *Sec.*). We computed the average objective value over the ten runs (Average), as well as the objective value of the worst and the best runs. They are expressed in terms of their difference in percentage with respect to Gurobi's best known solution computed as: $\%GUR = (ILS\ Obj - GUR\ Obj)/(GUR\ Obj)$.

Table 2 shows that the mathematical formulation was solved to optimality for all small and medium-sized instances in short computational times. As per the heuristic algorithm let us first discuss the results of the initialization stage of the algorithm, which is deterministic and hence reports the same value every run. Computational times are not reported as they were, in all the cases, negligible. For the small and medium-sized instances the initialization results are 5.53% above the optimum, with 8 out of the 27 instances solved to optimality at this stage. On the other hand, five instances are less than 5% above the optimal, and five instances are more than 10% above the optimum value, with the highest %GUR over 19%. The ILS shows a good performance. Indeed, it finds optimal solutions to seven more instances, and in eight instances, its average gap is reduced to less than 1% above the optimum. Overall, the average %GUR, over all runs and all instances, is reduced from 5.53% (initialization stage) to 0.46%, and the average CPU time required is of 11.7 seconds.

Table 2 – Numerical results produced for the Small and medium-sized instances

Instance			**Gurobi**			**Init. Stage**	**ILS - 10 runs**			
								Average	**Worst run**	**Best run**
Name	**N**	**P**	**Obj.**	**Sec.**	**Gap%**	**%GUR**	**Sec.**	**%GUR**	**%GUR**	**%GUR**
I01	2	3	270	0.0	0.00	**0.00**	0.7	**0.00**	**0.00**	**0.00**
I02	3	3	174	0.1	0.00	2.87	0.4	0.57	2.87	**0.00**
I03	2	4	358	0.4	0.00	**0.00**	0.5	**0.00**	**0.00**	**0.00**
I04	3	4	308	0.0	0.00	**0.00**	0.4	**0.00**	**0.00**	**0.00**
I05	3	6	404	0.4	0.00	1.73	1.4	1.04	1.73	**0.00**
I06	5	7	391	0.3	0.00	**0.00**	3.3	**0.00**	**0.00**	**0.00**
I07	5	7	383	0.4	0.00	**0.00**	3.4	**0.00**	**0.00**	**0.00**
I08	5	7	384	0.5	0.00	0.78	3.2	**0.00**	**0.00**	**0.00**
I09	4	7	352	0.5	0.00	19.89	0.8	**0.00**	**0.00**	**0.00**
I10	2	7	628	0.3	0.00	**0.00**	0.7	**0.00**	**0.00**	**0.00**
I11	4	7	716	0.2	0.00	**0.00**	1.4	**0.00**	**0.00**	**0.00**
I12	4	7	305	0.4	0.00	7.54	1.3	1.57	3.93	**0.00**
I13	4	8	815	0.4	0.00	0.49	2.6	0.15	0.49	**0.00**
I14	5	8	347	0.6	0.00	6.63	3.5	**0.00**	0.29	**0.00**
I15	5	8	461	0.2	0.00	11.93	2.1	**0.00**	**0.00**	**0.00**
I16	4	9	304	0.5	0.00	**0.00**	0.4	**0.00**	**0.00**	**0.00**
I17	4	9	313	0.5	0.00	10.22	3.8	1.79	5.43	**0.00**
I18	4	9	320	0.5	0.00	11.25	3.1	3.97	4.38	0.94
I19	6	9	512	0.4	0.00	16.99	5.1	**0.00**	**0.00**	**0.00**
I20	6	11	485	2.9	0.00	7.01	9.9	0.85	3.30	**0.00**
I21	9	13	412	37.2	0.00	3.88	21.2	**0.00**	**0.00**	**0.00**

I22	8	13	497	9.9	0.00	9.86	18.2	**0.00**	**0.00**	**0.00**
I23	7	14	1 058	3.8	0.00	6.33	20.3	0.67	0.95	**0.19**
I24	9	16	1 185	16.4	0.00	5.65	37.4	0.04	0.42	**0.00**
I25	9	16	1 167	2.0	0.00	9.68	25.7	0.07	0.09	**0.00**
I26	12	18	1 343	859.4	0.00	8.04	67.2	0.63	1.41	0.45
I27	11	21	1 928	726.8	0.00	8.51	79.2	0.94	1.04	0.52
Average				**61.7**	**0.00**	**5.53**	**11.7**	**0.46**	**0.98**	**0.08**

If we look at the best solutions produced over the ten executions, ILS was able to produce optimal solutions to 24 out of 27 instances, with an average gap of only 0.08%. To give an idea of the ILS's robustness, we also report the results produced by the "worst" run for each instance. On the worst run, ILS is still able to find 14 optimal solutions out of 27 instances, showing an average performance within 1% over optimality.

Let us now analyze the results produced for the largest instances. As it can be observed in Table 3, the solver Gurobi was unable to handle the larger instances efficiently. In fact, despite a computational time limit of 36000 seconds (10 hours), the optimality gap produced by Gurobi ranges from 48% to 100%, with an average of 86%. It is therefore very difficult to estimate the quality of the best solutions produced by Gurobi. Nonetheless, in Table 3 we will still use the best solutions produced by Gurobi as the baseline to evaluate the ILS's performance.

Table 3 – Numerical results produced for the Large-sized instances

Instance			**Gurobi**		**Init. Stage**	**ILS - 10 runs**			
							Average	**Worst run**	**Best run**
	N	**P**	**Obj.**	**Gap%**	**%GUR**	**Sec.**	**%GUR**	**%GUR**	**%GUR**
I28	17	24	2 088	47.73	27.68	364.1	0.58	1.15	0.10
I29	10	30	951	100.00	15.30	218.2	0.59	2.00	**-0.53**
I30	11	32	990	100.00	17.38	312.8	0.69	2.17	**-1.62**
I31	12	34	1 089	100.00	14.38	390.3	2.19	3.81	0.87
I32	17	35	1 786	76.08	9.41	720.3	**-3.71**	**-1.40**	**-4.93**
I33	13	36	1 147	100.00	12.08	488.9	**-0.91**	0.48	**-2.75**
I34	18	39	1 881	86.81	6.62	1 288.3	**-1.73**	**-0.72**	**-3.11**
I35	19	40	1 985	76.64	14.41	1 225.1	**-1.02**	**-0.20**	**-2.12**
I36	19	40	2 039	79.10	19.96	1 257.2	**-2.62**	**-1.42**	**-3.92**
I37	19	41	2 112	81.16	21.78	1 484.8	**-0.78**	1.28	**-2.18**
I38	17	50	1 170	100.00	11.20	3 365.0	**-4.57**	**-3.85**	**-4.79**
Average			**566.9**	**86.14**	**15.47**	**1 010.4**	**-1.03**	**0.30**	**-2.27**

Table 3 shows that the initialization stage is clearly not able to produce good quality solutions. Indeed, it produces solutions that are, on average, 15.47% above the best known solutions produced by Gurobi and, in the worst case, up to 27.68% worse than Gurobi. Let us now look at the results produced by 10 executions of the ILS algorithm. Considering the average results over the ten runs, ILS produced in seven out of 11 instances an average objective value that is better than the solver's solution, and over the 11 instances, it produces an improvement of 1.03% with respect to Gurobi's best known solutions. In addition, the time to execute 10 runs of the algorithm ranges from 218 up to 3365 seconds, which is, in the worst case, 10 times less than the time allotted to Gurobi. If we look at the best run over the ten runs, ILS was able to produce solutions that improved Gurobi in 9 out of 11 instances, with a best improvement of 4.93% and an average improvement of 2.27%. Finally, it is also worthy to note the good robustness of the algorithm. Indeed, the worst run over the ten produced, on average, solutions close to the ones produced by Gurobi.

We can therefore conclude that, when compared to a commercial solver, the ILS algorithm produces high quality solutions in reasonable computational time. Moreover, executing the algorithm 10 times contributes to improving its robustness, balancing efficiently quality of the solutions and computational effort.

6. Conclusion and Research Perspectives

This paper presents an extended version of the biomedical sample transportation problem (BSTP) and proposes an iterated local search (ILS) heuristic to tackle the large-sized instances that cannot be solved efficiently by a commercial solver. Three important features of this version make this work an interesting contribution to the VRP. Firstly, we deal with a problem where no time windows are imposed to visit the customers, but the pickups' hours are decided endogenously in the model to respect each SCC's service level requirement (precedence constraints). Secondly, we add to this synchronization the flexibility on SCCs' opening hours (tactical decision) that leads to better transportation plans but also increases the complexity of the problem. Thirdly, the daily routing decisions are directly affected by the synchronization constraints, but also by a maximum transportation time fixed by the most urgent sample in the route (which differs for a total route length limit).

Due to the fact that our partner, the MSSS, intends to plan real collection routes, an efficient algorithm is required to solve the problem in short computational time. We thus propose an ILS that after execute an *Initialization* stage, iterates between a *"soft" adaptive diversification* and a *hard diversification* stages. The proposed algorithm has been tested over 38 real instances from Quebec's laboratory network. Our numerical experiments have shown that permitting a small amount of flexibility on the opening hours leads to better routes in 19 out of 27 small instances, with improvements that can reach up to 16.59% with respect to the routes duration produced when no flexibility is allowed. As per the large instances, we can improve the transportation plan's quality by 19.75% for instance 28. We also find that, inside the time limit of the 10 hours, the model with flexible opening decisions identifies a better solution than the rigid model.

We were also able to show the good performance of the ILS algorithm, both in terms of solution quality and computational time. On average, ILS achieves a deviation of only 0.08% above the optimum for small and medium size instances, requiring only 11.7 seconds of computing. For the larger instances, and on average, it improves the solutions produced after 10 hours of computing by the Gurobi solver by 2.27%, doing it in a fraction of the computing time. This work contributes to the tactical planning of sample transportation at the MSSS and it supports their network reengineering process. In particular, it will help the network's stakeholders in their negotiation process to coordinate their activities, where fast and efficient solution approaches are required.

Our research is now shifting to the use of a matheuristic approach merging the exact formulation and approximated solution techniques. In addition, we must assess the robustness of the proposed solutions facing the practical uncertainties such as, for example, transportation times or delays in the samples' preparation that might affect the solutions produced by our model.

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), *Discovery Grants* Program [grants OPG 0293307 and OPG 0172633], the scholarship from the *International research internship - Strategic Clusters* Program, attributed to Ana María Anaya-Arenas, by the Fonds de recherche du Québec - Nature et technologies (FRQNT), and the financial support of the Ministry of Healthcare and Social Services of Quebec (MSSS). This financial support is gratefully acknowledged.

References

Amorim, P., Parragh, S. N., Sperandio, F. & Almada-Lobo, B. (2014). A rich vehicle routing problem dealing with perishable food: a case study. *Top, 22*(2), 489–508.

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., Asfar, H. M. & Prins, C. (2015). A metaheuristic for the biomedical sample transportation problem in the Province of Quebec. *Proceedings of the 11e Congrès International de Génie Industriel - CIGI 2015*, Quebec City, Canada.

Andersson, H., Duesund, J. M. & Fagerholt, K. (2011). Ship routing and scheduling with cargo coupling and synchronization constraints. *Computers & Industrial Engineering*, 61(4), 1107–1116.

Bakkehaug, R., Rakke, J. G., Fagerholt, K. & Laporte, G. (2016). An adaptive large neighborhood search heuristic for fleet deployment problems with voyage separation requirements. *Transportation Research Part C: Emerging Technologies*, 70, 129–141.

Baş, S., Carello, G., Lanzarone, E., Ocak, Z. & Yalçindag, S. (2016). Management of Blood Donation System: Literature Review and Research Perspectives. *Health Care Systems Engineering for Scientists and Practitioners* (pp. 121–132). Lyon, France.

Belhaiza, S., Hansen, P. & Laporte, G. (2014). A hybrid variable neighborhood tabu search heuristic for the vehicle routing problem with multiple time windows. *Computers & Operations Research*, 52, 269–281.

Braekers, K., Ramaekers, K. & Van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99, 300–313.

Bräysy, O. & Gendreau, M. (2005a). Vehicle routing problem with time windows, Part II: Metaheuristics. *Transportation science*, 39(1), 119–139.

Bräysy, O. & Gendreau, M. (2005b). Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation science*, 39(1), 104–118.

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.

Coelho, L. C., Cordeau, J.-F. & Laporte, G. (2012). Consistency in multi-vehicle inventory-routing. *Transportation Research Part C: Emerging Technologies*, 24, 270–287.

Coelho L. C., Renaud J. & Laporte G. (2016). Road-based goods transportation : A survey of real-world logistics applications from 2000 to 2015. *INFOR : Information Systems and Operations Research*, 54, 79-96.

Cordeau, J.-F. & Laporte, G. (2007). The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1), 29–46.

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 & Operations Research*, 35(9), 3034–3048.

Doerner, K. F. & Hartl, R. F. (2008). Health care logistics, emergency preparedness, and disaster relief: new challenges for routing problems with a focus on the Austrian situation. The vehicle routing problem: latest advances and new challenges, (pp. 527–550), College Park, MD, USA.

Dohn, A., Rasmussen, M. S. & Larsen, J. (2011). The vehicle routing problem with time windows and temporal dependencies. *Networks*, 58(4), 273–289.

Drexl, M. (2012). Synchronization in vehicle routing-A survey of VRPs with multiple synchronization constraints. *Transportation Science*, 46(3), 297–316.

Dumas, Y., Desrosiers, J. & Soumis, F. (1991). The pickup and delivery problem with time windows. *European journal of operational research*, 54(1), 7–22.

El Hachemi, N., Gendreau, M. & Rousseau, L.-M. (2013). A heuristic to solve the synchronized log-truck scheduling problem. *Computers & Operations Research*, 40(3), 666–673.

Favaretto, D., Moretti, E. & Pellegrini, P. (2007). Ant colony system for a VRP with multiple time windows and multiple visits. Journal of Interdisciplinary Mathematics, 10(2), 263–284.

Fikar, C. & Hirsch, P. (2015). A matheuristic for routing real-world home service transport systems facilitating walking. Journal of Cleaner Production, 105, 300–310.

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.

Goel, A. & Meisel, F. (2013). Workforce routing and scheduling for electricity network maintenance with downtime minimization. European Journal of Operational Research, 231(1), 210–228. Elsevier.Haddadene, S. R. A.,

Huang, S.-H. & Lin, P.-C. (2015). Vehicle Routing-Scheduling for Municipal Waste Collection System under the “Keep Trash off the Ground” Policy. Omega, 55, 24–37.

Ioachim, I., Desrosiers, J., Soumis, F. & Bélanger, N. (1999). Fleet assignment and routing with schedule synchronization constraints. European Journal of Operational Research, 119(1), 75–90.

Kergosien, Y., Ruiz, A. & Soriano, P. (2014). A routing problem for medical test sample collection in home health care services. Proceedings of the 22th International Conference on Health Care Systems Engineering (pp. 29–46), Milan, Italy.

Labadie, N., Prins, C. & Yang, Y. (2014). Iterated Local Search for a Vehicle Routing Problem with Synchronization Constraints. ICORES (pp. 257–263), Angers, France.

Labadie, N. & Prodhon, C. (2016). A GRASP x ILS for the vehicle routing problem with time windows, synchronization and precedence constraints. Expert Systems with Applications, 66, 274–294.

Lahyani, R., Khemakhem, M. & Semet, F. (2015). Rich vehicle routing problems: From a taxonomy to a definition. European Journal of Operational Research, 241(1), 1–14.

Lim, A., Zhang, Z. & Qin, H. (2016). Pickup and delivery service with manpower planning in hong kong public hospitals. Transportation Science, 51(2), 688–705.

Liu, R., Xie, X., Augusto, V. & Rodríguez, 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.

Mankowska, D. S., Meisel, F. & Bierwirth, C. (2014). The home health care routing and scheduling problem with interdependent services. Health care management science, 17(1), 15–30.

McDonald, J. (1972). Vehicle Scheduling-A Case Study. Operational Research Quarterly, 433–444.

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, 58–67.

Norstad, I., Fagerholt, K., Hvattum, L. M., Arnulf, H. S. & Bjørkli, A. (2015). Maritime fleet deployment with voyage separation requirements. Flexible Services and Manufacturing Journal, 27(2-3), 180–199.

Rasmussen, M. S., Justesen, T., Dohn, A. & Larsen, J. (2012). The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. European Journal of Operational Research, 219(3), 598–610.

Reinhardt, L. B., Clausen, T. & Pisinger, D. (2013). Synchronized dial-a-ride transportation of disabled passengers at airports. European Journal of Operational Research, 225(1), 106–117.

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

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

Tricoire, F., Romauch, M., Doerner, K. F. & Hartl, R. F. (2010). Heuristics for the multi-period orienteering problem with multiple time windows. Computers & Operations Research, 37(2), 351–367.

Vidal, T., Crainic, T. G., Gendreau, M. & Prins, C. (2013). A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. Computers & Operations Research, 40(1), 475–489.

Vidal, T., Crainic, T. G., Gendreau, M. & Prins, C. (2014). A unified solution framework for multi-attribute vehicle routing problems. European Journal of Operational Research, 234(3), 658–673.

Vilhelmsen, C., Lusby, R. M. & Larsen, J. (2014). Tramp ship routing and scheduling with voyage separation requirements. OR Spectrum, 1–31.

Xu, D., Li, K., Zou, X. & Liu, L. (2017). An unpaired pickup and delivery vehicle routing problem with multi-visit. Transportation Research Part E: Logistics and Transportation Review, 103, 218–247.

Yi, J. (2003). Vehicle routing with time windows and time-dependent rewards: A problem from the American Red Cross. Manufacturing & Service Operations Management, 5(1), 74–77.

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

Appendices

The dependency between the pickup times at a given SCC creates interdependence between the routes, which forces us to verify the feasibility of every new solution generated. This additional difficulty to explore the neighbourhood of a solution through a local search has also been pointed out in other applications (e.g. Rousseau et al., 2013). We thus implement two general mechanisms to mitigate this drawback. First, to reduce the need for feasibility checks caused by the interdependence between transportation requests, we compute time bounds for each visit in a given solution. These bounds define a "time window" inside which each transportation request must be performed to guarantee that the previous and next visits to the SCC remain feasible. The second consists in implementing the concatenation techniques described in Vidal et al. (2014) to efficiently evaluate cost and feasibility whenever a change is introduced in a route, adapting them to our problem constraints. These two mechanisms are thoroughly explained in the forthcoming Appendix 1 and Appendix 2.

Appendix 1 - Time Bounds on an SCC's Transportation Requests

We propose to use time bounds for SCC's transportation request in order to reduce the interdependency in the visits' hours and explore the neighbourhood faster and efficiently. For a given SCC c_g, the time window for the opening hours combined with the maximum timespan between pickups defines time bounds for the collections to be performed at c_g. This first rough calculation of the possible visits time can be very large and it might lead to infeasibility for the SCC's service requirement. However, once the decision regarding the visit time is made, time windows are shrunk and reflect the "real flexibility" of the pickups, without affecting the other requests of that center. Hence, to reduce the interdependency between pickups during the construction or the LS evaluation process, we estimate time bounds for each transportation request based on the current (either partial or complete) solution. For index notation simplicity, we present an example of the time window computation for SCC c_1. We define $[\alpha_k - \beta_k]$ as the earliest and latest possible time at which pickup v_k (with $k = 1, \dots, |P_1|$) can be made without affecting the other pickups of SCC c_1. Let u_k be the service time of any k transportation requests of SCC c_1 in a solution s. First, the opening and closing time windows are reduced by the service time of the first and the penultimate requests. The service time for the first requested pickup of SCC c_1 (i.e. u_1) might delay the earliest opening hour (to find a new limit, named e'_1) and service time of the penultimate pickup ($u_{|P_1|-1}$) might advance the latest possible closing (thus opening) hour defining l'_1 (see Figure 4). Precisely, in a given solution s, the earliest opening for SCC c_1 is defined by equation (18) and latest opening by equation (19):

$$e'_1 = max\{e_1; u_1 - \Delta^1_{max}\} \tag{18}$$

$$l'_1 = min\{l_1; u_{|P_1|-1} + \Delta^1_{max} - O_1\} \tag{19}$$

The new time window for starting and ending of the collection period of SCC c_1, affects the time window for the last pickup, as well as the first and penultimate one, as follows:

$$\alpha_{|P_1|} = e_1' + O_1 \quad (20)$$

$$\beta_{|P_1|} = l_1' + O_1 + \varphi_1 \quad (21)$$

$$\beta_1 = l_1' + \Delta_{max}^1 \quad (22)$$

$$\alpha_{|P_1|-1} = e_1' + O_1 - \Delta_{max}^1 \quad (23)$$

For instance, it is known that the last pickup of the day has to be performed after the ending of the collection period (b_1) of the SCC c_1 and no later than φ_1 minutes after. Therefore, the earliest service time for the last pickup (20) is the earliest possible end of the collection period, thus, earliest opening plus the collection period ($e_1' + O_1$). The end of its time window (21) is the latest possible closing plus φ_1. The latest service time for the first pickup schedule is fixed by adding the maximum timespan to the latest possible opening of the SCC c_1 (22), and the earliest possible time service for the penultimate pickup is fixed by the earliest end of the collection period minus Δ_{max}^1 (23). Needless to say, the service time of pickup v_j (u_j) sets the latest possible service time for v_{j+1} (24) and the earliest possible service time for v_{j-1} (25).

$$\beta_{j+1} = u_j + \Delta_{max}^1 \quad (24)$$

$$\alpha_{j-1} = u_j - \Delta_{max}^1 \quad (25)$$

These calculations are illustrated in Figure 4. For an SCC that requests only one pickup, time windows are always the same (see constraints 6 and 10).

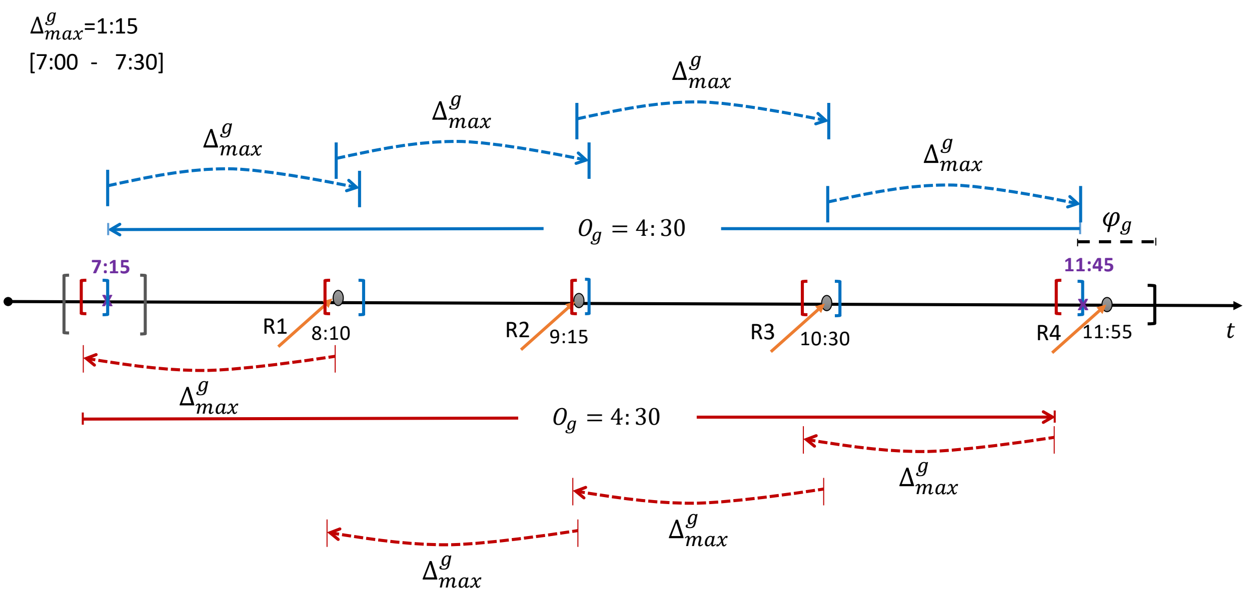

Figure 4– Time bounds estimation for SCC c_1 in solution s.

This approach eliminates the interdependency for any two consecutive pickups during the evaluation of moves. However, due to the fact that the opening and closing times are also flexible inside a time window, there is still interdependency between the first, penultimate and last pickup of each client. Consider for instance the case of SCC c_1 in a solution s. c_1 has an opening time window from $[7{:}00 - 7{:}30]$, $O_1 = 4{:}30$ and $\Delta^1_{max} = 1{:}15$. The SCC's opening is fixed to $7{:}15$ and its four pickups are scheduled in four routes (R1 to R4), at $8{:}10, 9{:}15, 10{:}30$ and $11{:}45$. Consider that a movement in R1 could delay the arrival to the SCC c_1 to $8{:}45$. In order to validate if this delay still leads to a feasible solution, the earliest and latest opening hours must be delayed to 7:30 (to respect the Δ^1_{max}), also delaying the end of the collection period to noon. However, this will imply that the last pickup is no longer feasible, because it is performed before the end of the collection period (as illustrated in Figure 5). Likewise, an advance in service time of the penultimate pickup might result in infeasibility for the first pickup. This last dependency cannot be avoided and it is checked for every move considered that affects any of those three requests (for every SCC). If infeasible, the movement will be refused.

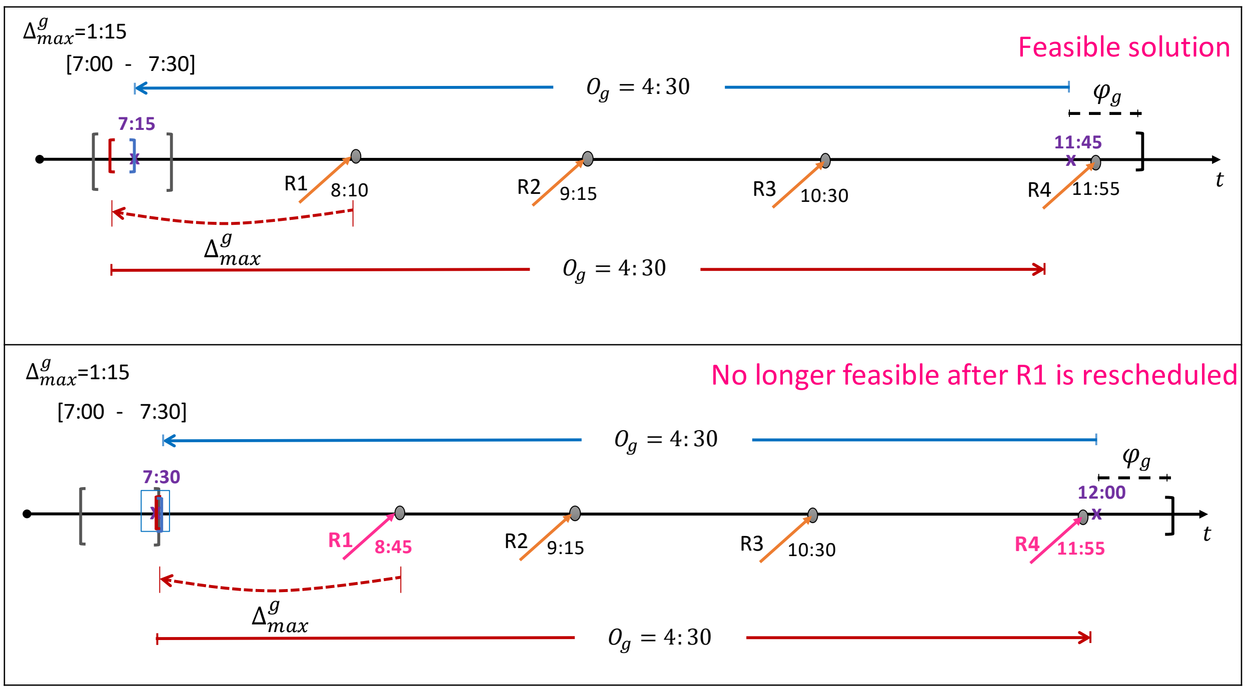

Figure 5– Movement in R1 leading to an unfeasible solution.

Appendix 2 - Cost and Feasibility Check

In order to evaluate a movement efficiently in the construction of the LS procedure, we implement the sequence concatenation technique proposed by Vidal et al. (2014). The basic idea is to define any solution (i.e. set of routes) as the combination of different sequences of visits. Moreover, edge exchanges and node relocations are finally the recombination of known sequences. Estimating significant information to describe the sequences and evaluate its concatenation allows us to evaluate all moves in constant time. However, once a move is implemented, the entire solution information has to be updated.

The significant information that is used to characterize any subsequence σ of the BSTP are:

- the total service time $T(\sigma)$, i.e. the sum of travel and service time,
- the earliest completion time of a sequence $E(\sigma)$,
- the latest start time $L(\sigma)$,
- the minimal duration of the sequence $D(\sigma)$,
- and the feasibility statement $F(\sigma)$.

In addition, in order to control the return of all the samples to the lab before they perish, we compute $Tlim(\sigma)$ as the maximum transportation time available when the vehicle leaves the last SCC in the sequence σ, so all its samples arrive to lab on time. For a sequence of a single visit i ($\sigma_0 = \{v_i\}$) we define:

$$T(\sigma_0) = D(\sigma_0) = \tau_i \tag{26}$$

$$E(\sigma_0) = \alpha_i + \tau_i \tag{27}$$

$$L(\sigma_0) = \beta_i \tag{28}$$

$$Tlim(\sigma_0) = T_{max}^{i} - \tau_i \tag{29}$$

$$F(\sigma_0) = true \tag{30}$$

We refer the interested reader on the equations used to compute the information for a concatenation of sequence for $T(\sigma), E(\sigma)$, $L(\sigma)$ to Vidal et al. (2014). Now, remember that waiting time is allowed before serving a customer. Its estimation (ΔWT), as well as the one for its minimum duration $D(\sigma)$, are defined in Vidal, Crainic, Gendreau & Prins, (2013).

We propose in (31) and (32) how $Tlim(\sigma_1 \oplus \sigma_2)$ and $F(\sigma_1 \oplus \sigma_2)$ have to be computed.

$$Tlim(\sigma_1 \oplus \sigma_2) = \; Min\left\{Tlim(\sigma_1) - \Delta WT - t_{\sigma_1(|\sigma_1|),\sigma_2(1)} - D(\sigma_2), Tlim(\sigma_2)\right\} \tag{31}$$

$$\begin{gathered} F(\sigma_1 \oplus \sigma_2) \equiv F(\sigma_1) \wedge F(\sigma_2) \; \wedge \left(E(\sigma_1) + t_{\sigma_1(|\sigma_1|)\sigma_2(1)} \leq L(\sigma_2)\right) \wedge \\ (\, Tlim(\sigma_1) - \Delta WT - t_{\sigma_1(|\sigma_1|),\sigma_2(1)} - D(\sigma_2) \; \geq t_{\sigma_2(|\sigma_2|),0}) \end{gathered} \tag{32}$$