Optimizing specimen collection for processing in clinical testing laboratories

E. Yücel, F.S. Salman, E.S. Gel, E.L. Örmecci, A. Gel

Abstract

We study the logistics of specimen collection for a clinical testing laboratory that serves sites dispersed in an urban area. The specimens that accumulate at the customer sites throughout the working day are transported to the laboratory for processing. The problem is to construct and schedule a series of tours to collect the accumulated specimens from the sites throughout the day. Two hierarchical objectives are considered: (i) maximizing the amount of specimens processed by the next morning, and (ii) minimizing the daily transportation cost. We show that the problem is NP-hard and formulate a linear Mixed Integer Programming (MIP) model to solve the bicriteria problem in two levels. We characterize properties of optimal solutions and develop a heuristic approach based on solving the MIP model with additional constraints that seeks for feasible solutions with specific characteristics. To evaluate the performance of this approach, we provide an upper bounding scheme on the daily processed amount, and develop two relaxed MIP models to generate lower bounds on the daily transportation cost. The effectiveness of the proposed solution approach is evaluated using realistic problem instances. Insights on key problem parameters and their effects on the solutions are extracted by further experiments.

Keywords:
OR in health services
Transportation
Logistics of clinical testing laboratories

1. Introduction

Clinical laboratory testing is an essential element in the delivery of healthcare services. Healthcare professionals use laboratory tests to assist in the detection, diagnosis, monitoring and treatment of diseases and other medical conditions. Clinical testing is performed on bodily fluids, such as blood and urine, and is usually outsourced to specialized clinical testing companies by hospitals and healthcare professionals. These companies analyze the specimens on special-purpose medical testing equipment, and compile the results for each patient in a report, which is then sent back to the patient’s healthcare provider.

It is estimated that lab testing has an impact on over 70% of medical decisions (Knowledge Source Inc., 2008). It is essential that clinical laboratories provide error-free processing and reporting, as well as reliable service with short turn-around times of test results. Most clinical testing companies have strict agreements on turn-around time, which is generally required to be less than one business day, and sometimes as short as a few hours for urgent requisitions. Achieving a high level of customer service can be possible only if the collection and delivery of specimens to the laboratory is planned and executed efficiently. Additionally, clinical testing companies cannot prioritize or refuse to serve customers based on profit concerns to maintain market competitiveness and due to ethical obligations.

Clinical testing companies are faced with the following daily operations. Clinical specimens are obtained from patients at various sites, such as physician offices, hospitals, and patient service centers operated by the companies. The specimens are then transferred to a central processing facility to be analyzed. The collection of the specimens from the sites is achieved by a fleet of vehicles that visit each site a number of times every day. The sites are grouped into several regions, each of which is serviced by one of the vehicles. The drivers visit the sites according to a predetermined sequence, except for situations when an urgent requisition arises. Once the specimens get to the processing facility, the requisition orders are logged into a database system for tracking and billing of the orders. The specimens are then processed on highly-automated testing equipment that runs at a predetermined rate. Finally, the analysis results are compiled in a report.

Clinical testing companies are under increasing pressure to provide service at lower cost, and hence, they constantly seek ways to reduce their operational costs, even though their first priority is still to provide reliable and accurate results with a fast turn-around time. An important determinant of the operational costs is the effective utilization of the high-tech processing equipment and skilled labor, which in turn is affected by the way the specimens are transported from the sites to the laboratory. In particular, the arrival rate of specimens to the laboratory has to match (or exceed)
the processing rate so that starvation of the processing resources can be avoided.

We consider the problem of transporting specimens from a number of geographically-dispersed sites to the processing facility of a clinical testing company. The fundamental decisions to be optimized can be represented by the schedule (i.e., timing) of visits to each site throughout the day. This involves designing tours to be conducted by a vehicle throughout the day, which requires the determination of both the sequence of sites to visit and their timings. This problem is significantly different from a routing problem where the only decision of interest is the sequence in which the sites will be visited by each vehicle. Another major difference is that we consider the number of specimens processed as a prioritized objective, in addition to the transportation cost, since collection strategies that only minimize transportation cost may result in excessive idle times at the processing facility. On the other hand, keeping the processing unit busy may dictate the use of more frequent, and possibly more costly tours. Hence, transportation decisions should be made with the consideration of the input that they provide to the processing system.

In this paper, we define a single-vehicle deterministic version of this problem as the Collection for Processing Problem (CfPP). Specifically, we define CfPP as the problem of designing tours to match collected workload with the processing capacity at the processing facility. CfPP involves two hierarchical objectives. The first-level objective is to maximize the daily processed amount, whereas the second-level objective is the minimization of transportation costs, which are assumed to be directly proportional to the distance traveled by the vehicle. In particular, we seek a solution that minimizes the transportation costs while maintaining the maximized daily processed amount obtained by optimizing only with respect to the first level objective.

Clinical specimen collection was first addressed by McDonald (1972) in a case study, where the objective was to minimize the total traveling time. The problem is formulated as a Vehicle Routing Problem (VRP), and a heuristic algorithm is proposed for its solution. Revere (2004) also discusses a case study on a business process re-engineering project for a laboratory courier service. The goal is to minimize both the laboratory courier and staffing costs. The problem is divided into two sub-problems, each dedicated to one of the two objectives. For the objective of minimizing laboratory courier cost, a traveling salesman model is used. An integer programming formulation is developed to minimize staffing costs. Differently from our study, neither of these studies considers the accumulation of specimens over time.

A related problem appears in the domain of blood collection. CfPP has two key common features with blood collection problems studied before: (i) units to be collected accumulate over time and (ii) there is a deadline for the completion of processing. Operational efficiency of the blood collection process has been studied since the 1970s (e.g., Dumais and Rabinowitz, 1977; Or, 1976), albeit, without vehicle routing. Yi and Scheller-Wolf (2003) is the first paper that incorporates vehicle routing aspects into this problem, through a real life application at the American Red Cross. Recently, Doerner et al. (2008) studied logistics of the blood program in the Austrian Red Cross as a vehicle routing problem with multiple independent time windows for each customer. Although these studies consider accumulation over time in the sites, none of them considers the impact of finite processing capacity.

CfPP is a routing problem with multiple tours, accumulation over time, a finite processing capacity, a processing deadline, and multiple prioritized objectives. To the best of our knowledge, ours is the first study that addresses this problem, although the following well-studied routing problems possess similar characteristics that deserve to be discussed.

The Team Orienteering Problem (TOP) aims to route a number of vehicles through a set of nodes each of which contains a fixed reward (i.e., no accumulation by time), in order to maximize the total collected reward while ensuring that all vehicles return to the pre-determined nodes within a given time limit. Two exact algorithms have been proposed for the TOP; one that uses column generation (Butt and Ryan, 1999) and another one that uses a branch-and-price algorithm (Boussier et al., 2007), to solve small-to moderate-sized instances. A survey of heuristic methods is provided in Vansteenwegen et al. (2009).

The Inventory Routing Problem (IRP) is concerned with the distribution of products from a single facility to a set of nodes to satisfy customer demand over a given planning horizon. The objective of IRP is to minimize the operating costs, which consists of transportation and inventory holding costs (Moin and Salhi, 2007). Differently from the IRP, in the CfPP, accumulated items do not incur holding costs and the primary objective is to maximize the daily processed amount. Both IRP and CfPP combine a temporal element (node visit times) with a spatial element (vehicle routing), but there is no consideration of processing in IRP.

Inbound logistics problems arising in just-in-time or lean production systems are concerned with coordinating the material inflow with the production rate, similar to the CfPP. In a recent study, Ohlmann et al. (2008) consider vehicle routing with time-windows to collect components from suppliers in a lean production system. The objective is to minimize the total traveling cost subject to inventory limitations.

While the above listed family of problems have similarities to CfPP, we can summarize the basic differences of CfPP from these problems as follows: (i) The main objective is to maximize the processed amount by a deadline. Operational costs are of secondary concern since a case of unprocessed items will have serious consequences, e.g., the laboratory may lose the business of the corresponding site. (ii) The processing rate at the processing facility is explicitly modeled. (iii) The workload at each site accumulates over time, so that the collected amounts depend on the node visit times. (iv) The tours have to be designed simultaneously due to the significant dependency between the tours. In every tour, the return time of the vehicle to the processing facility, as well as the collected amount affect the processed amount up to the return time of the vehicle from the next tour. Furthermore, due to accumulation, the amount collected from a node in a tour depends on the last visit time of that node.

In this paper, we prove that CfPP is NP-hard, and introduce a Mixed Integer Programming (MIP) formulation for the problem. We characterize some properties of optimal solutions and identify rules to eliminate feasible solutions that are likely to be suboptimal. Based on these results, we develop a heuristic approach that solves the MIP model with additional constraints that reduce the solution space. To evaluate the performance of this approach, we provide methods to bound the two objectives. For the challenging task of generating strong lower bounds on the daily transportation cost for a given daily processed specimen level, we derive valid inequalities and develop two relaxed MIP models that rely on these inequalities and an alternative flow formulation. We conduct computational experiments on realistic test instances to demonstrate the effectiveness of our approach. We also extract insights on key problem parameters and their effects on the solutions by further experiments.

The remainder of this paper is organized as follows. In Section 2, we provide the problem definition and some propositions. Section 3 derives an upper bound on the daily processed amount. In Section 4, we provide an MIP formulation, and describe two relaxations of the MIP to generate lower bounds on daily transportation cost for a given daily processed amount in Section 5. Section 6 presents our approach to find a feasible solution from the
2. Problem description and notation

In this section, we provide a formal description of the CFP. We assume that items accumulate between time 0 and time \( \tau_p \), and processing can be performed at the processing facility between time 0 and time \( \tau_f \). The problem is defined on a directed graph \( G = (N,A) \), where \( N = \{0,1,2,\ldots,n\} \) is the node set and \( A = \{(i,j) | i,j \in N\} \) is the arc set. Each arc \((i,j)\) is associated with non-negative travel time \( t_{ij} \) and distance \( d_{ij} \). We assume that both the distances and the travel times are metric. Node 0 represents the processing facility, and \( N = N \setminus \{0\} \) is the set of sites. Items accumulate at each site \( i \in N \) at a constant and known rate of \( \lambda_i \) units per unit time. Without loss of generality, we label the sites so that \( \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n \). Items are processed at the processing facility at a constant and known rate of \( \mu \) units per unit time. We let \( I_i(t) \) represent the amount of items at site \( i \) at time \( t \), where \( i \in N \), and \( g_i(t) \) denote the amount of items waiting in the queue or undergoing processing at the processing facility at time \( t \).

We consider the decisions regarding a single vehicle dedicated to serve the node set, \( N \). The vehicle is assumed to have sufficiently large capacity to transfer the daily volume of all sites, which is a realistic assumption for clinical laboratory logistics. We assume that the vehicle is positioned at the processing facility at time 0.

Appendix A summarizes all of the notation used throughout the paper.

Let us denote the set of feasible solutions for a CFP instance by \( G \). For a solution \( g \in G \), let \( P \) represent the daily processed amount, and \( C \) represent the daily transportation cost, which is directly proportional to the distance traveled. The objective of CFP is first to maximize the daily processed amount over the set \( G \), and then to select a solution with minimum transportation cost among the ones with maximum daily processed amount. We call such solutions optimal.

For a CFP instance, the daily processing capacity of the processing unit, i.e., the maximum amount of items that can be processed between times 0 and \( \tau_f \), is equal to \( \tau_f \mu \). The daily accumulated amount, i.e., the amount of items accumulated at all sites in one day, is equal to \( \tau_f \sum_{i \in N} \lambda_i \), all of which must be transported to the processing facility.

**Definition 1.** The workload level, \( \alpha \), for a CFP instance is defined as the ratio of the daily accumulated amount to the daily processing capacity. That is,

\[
\alpha = \frac{\tau_f \sum_{i \in N} \lambda_i}{\tau_f \mu}.
\]

In systems with heavy workload (i.e., large \( \alpha \)), it may not be possible to process all the daily volume by \( \tau_f \). On the other hand, when the workload is relatively low (i.e., small \( \alpha \)), there may be many solutions that can process all of the daily volume on time, in which case minimization of the daily transportation cost gains importance.

Let us consider a small illustrative example. Suppose that \( \tau_f = 540 \) and \( \tau_f = 1200 \) minutes, the processing rate is \( \mu = 100 \) items per hour and there are four sites with accumulation rates of \( \lambda_1 = 7 \), \( \lambda_2 = 30 \), \( \lambda_3 = 50 \), and \( \lambda_4 = 106 \) items per hour. Symmetric distances are given as \( d_{01} = 36 \), \( d_{02} = 24 \), \( d_{03} = 25 \), \( d_{04} = 15 \), \( d_{12} = 15 \), \( d_{13} = 22 \), \( d_{14} = 15 \), \( d_{23} = 18 \), \( d_{24} = 10 \), and \( d_{34} = 11 \) kilometer. For simplicity, we take \( t_{ij} = d_{ij} \) assuming that the speed of the vehicle is 1 kilometer per minute. In this CFP instance, the daily volume accumulated in all sites is \( \tau_f \sum_{i \in N} \lambda_i = 1737 \), whereas the daily processing capacity is \( \tau_f \mu = 2000 \). Therefore, \( \alpha = 1737/2000 = 0.869 \). This workload level indicates that it may be possible to process all items depending on the transportation decisions.

In systems where all of the daily volume cannot be processed on time, the unprocessed items that remain at the end of processing (i.e., at time \( \tau_f \)) may be backlogged to be processed in the next day, i.e., \( g_i(0) > 0 \) for the next day. In this paper, as we solve a single-day problem, we assume \( g_i(0) = 0 \). Section 7.2.5 analyzes a multi-day problem with \( g_i(0) > 0 \).

A solution for a CFP instance has two components. The first component is the sequence of nodes to be visited throughout the day. In other words, a solution can be represented as a concatenation of tours, where each tour starts and ends at the processing facility and visits a subset of sites in \( N \). Since the vehicle can wait at a node, the second component of a CFP solution provides the waiting times at the visited nodes. Hence, the travel times between nodes and waiting times at the nodes determine the node visit times. In order for a solution to be feasible, it should visit all sites after time \( \tau_f \) to collect all the remaining accumulated items, and should be completed by time \( \tau_f \). We note that the vehicle is allowed to be touring at time \( \tau_f \). For our example, if the tours performed by the vehicle throughout the day are \( 0-3-4-1-0, 0-4-2-0, 0-3-0, \) and \( 0-4-1-2-0 \), then the corresponding sequence is \( 0-3-4-1-0-4-2-0-3-0-4-1-2-0 \). There might be more than one feasible solution for this sequence. One such solution is \( 0-3-4-1-0-153-4-2-0(251)-3-0-4-1-2-0 \), where the numbers in parentheses denote the waiting times in minutes at the corresponding nodes. The vehicle waits only at the processing facility in this solution.

An optimal solution for our example is \( 0-74.7-4-3-0(142.1)-4-3-0-221.2-3-4-1-2-0 \), with \( P = 1737 \) and \( C = 182 \). In this solution, all of the daily accumulated amount is processed. The timeline representation of the solution and the amount of unprocessed items at the processing facility (i.e., \( g_i(t) \) in time interval \( [0, \tau_f] \)) are illustrated in Fig. 1. We see that the processor remains idle during the first tour and for a short interval during the second tour.

We next investigate the problem difficulty and certain properties of optimal solutions.

**Proposition 2.1.** CFP is NP-hard. (A proof via reduction from the Traveling Salesman Problem (TSP), which is shown to be NP-hard (Golden et al., 1987) is given in Appendix C available as part of the Online Supplement.)

The set of feasible solutions, \( G \), may be extremely large even for problems with limited number of nodes (e.g., \( n \geq 10 \)). In order to simplify the search for optimal solutions in a large feasible solution space, we investigate the structure of optimal solutions. The following two propositions allow us to concentrate on certain types of solutions.

**Proposition 2.2.** There exists an optimal solution in which the vehicle does not wait at any collection site. (Proof is provided in Appendix D1 available as part of the Online Supplement.)

**Proposition 2.3.** There exists an optimal solution in which sites are not visited more than once in any tour. (Proof is provided in Appendix D2 available as part of the Online Supplement.)

Since any solution that does not satisfy these properties can be converted to one with these properties without any loss of performance, we search only for such solutions.

3. An upper bound on daily processed amount

We next provide a strong upper bound for the daily processed amount. We use this upper bound to analyze the performance of...
our solution approach with respect to the first-level objective of maximizing daily processed amount.

The daily processed amount, \( P \), cannot exceed the daily accumulated amount or the daily processing capacity. Therefore, \( \min \{ t_s \sum_i N_i, t_f \} \) provides an upper bound for \( P \). However, this bound may be quite loose, since it does not consider the time required for the transportation of items from the sites to the processing facility. In order to obtain a tighter bound, we relax the constraint on the number of vehicles and define \( UB_p \) as the processed amount until \( t_f \).

When there is no limit on the number of vehicles available, a vehicle can be dispatched to each site every time unit. Then, the first delivery of items to the processing facility occurs at time \( (t_0 + t_0) \). If \( t_0(0) = 0 \), then the processing facility will be idle until time \( \min \{ t_0 + t_0 \} \). Since the accumulation of items at the sites ends at time \( t_f \), the latest arrival of items to the processing facility occurs at time \( t = t_f + \max_i (t_0) \) from the farthest site. We give a pseudocode to calculate \( UB_p \) in Algorithm 1.

**Algorithm 1. UBp Algorithm**

```plaintext
Input: \( N_i, t_i, t_f \)
Output: UBp
1: Set UBp ← 0, t0 ← 0.
2: for \( i \in \{1, 2, \ldots, \tau_f\} \) do
3: Set \( \gamma ← 0 \). // \( \gamma \) denotes the amount of items that arrive at time \( t \).
4: for \( i \in N \) do
5: if \( t = t_0 + t_0 \) then
6: Set \( \gamma \leftarrow \gamma + t_0 + t_0 \). // The first item arrival from node \( i \).
7: else if \( t_0 + t_0 < t < t_f + t_f \) then
8: Set \( \gamma \leftarrow \gamma + t \).
9: else
10: // Do nothing: no item arrival from node \( i \).
11: end if
12: end for
13: Set \( P \leftarrow \min(t_0, t) \). // Calculate the amount of processed items during time \( t \).
14: Set \( t_0 \leftarrow t_0 + \gamma - P \). // Update the amount of unprocessed items in the processing facility.
15: Set UBp ← UBp + P.
16: end for
```

We can demonstrate how \( UB_p \) improves the trivial bound \( UB_p = \min \{ t_s \sum_i N_i, t_f \} \) using the previously discussed example, where \( \alpha = 0.869 \). We also analyze the same system for \( \alpha \) values of 0.8, 1, and 1.2 by changing \( \mu \) accordingly (see Table 1). According to the results, \( UB_p \) improves \( UB_p \) around 3% when \( \alpha \gg 1 \), even for this small problem instance.

### Table 1

<table>
<thead>
<tr>
<th>( \alpha )</th>
<th>( \sum_i N_i t_f )</th>
<th>( \mu t_f )</th>
<th>UBp</th>
<th>UBp</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>1737</td>
<td>2171</td>
<td>1737</td>
<td>1737</td>
</tr>
<tr>
<td>0.869</td>
<td>1737</td>
<td>2000</td>
<td>1737</td>
<td>1737</td>
</tr>
<tr>
<td>1</td>
<td>1737</td>
<td>1737</td>
<td>1737</td>
<td>1692</td>
</tr>
<tr>
<td>1.2</td>
<td>1737</td>
<td>1448</td>
<td>1448</td>
<td>1410</td>
</tr>
</tbody>
</table>

4. **The mathematical model**

In this section, we develop an MIP model that determines the number of tours to be performed by the vehicle until time \( t_f \) as well as the start and end times of the tours throughout the day. It also selects the sites to be visited and their sequence in each tour. The number of tours that the vehicle can perform between times 0 and \( t_f \) is, naturally, bounded. Accordingly, we assume that the vehicle performs at most \( \kappa_1 \) tours during \( [0, t_f] \) and at most \( \kappa_2 \) tours during \( [t_{r}, t_f] \), where \( \kappa_1 \) and \( \kappa_2 \) are parameters to be determined by the decision makers. It is sufficient to visit a site just once after \( t_f \), as accumulation stops at this time. Hence, a natural bound on \( \kappa_2 \) is \( \kappa_2 \leq n \). However, the actual number of tours is determined by the model, since we allow empty tours in the formulation.

To calculate the accumulated amount at each node, we keep track of the last visit time of each node. Moreover, we need to know whether a node is visited before time \( t_f \) or not since accumulation stops at \( t_f \). The model ensures that all accumulated items are collected by the vehicle. The model allows the vehicle to wait only at the processing facility before \( t_f \) due to **Proposition 2.2**, and does not allow visiting a site more than once in a tour due to **Proposition 2.3**. We define the parameters, decision variables and constraints as follows.
Decision variables

\( X_{ijk} \) Binary variable indicating if node \( j \) is visited immediately after node \( i \) in tour \( k \) (\((i, j) \in A, k = 1, 2, \ldots, k_1 + k_2 \)). If tour \( k \) is an empty tour, then \( X_{00k} = 1 \) and \( X_{00k} = 0, X_{ijk} = 0, \forall i, j \in N \).

\( Y_{ik} \) Binary variable indicating if node \( i \) is visited in tour \( k \) (\( i \in N, k = 1, 2, \ldots, k_1 + k_2 \)).

\( T_k \) Visit time of node \( i \) at tour \( k \) (\( i \in N^+, k = 1, 2, \ldots, k_1 + k_2 \)). If node \( i \) is not visited in tour \( k \), it denotes the last visit time of node \( i \) before tour \( k \) (i.e., \( T_{ik} = T_{ik-1} \)). \( T_{0k} \) denotes the starting time of tour \( k \).

\( R_k \) Return time of the vehicle from tour \( k \) to processing facility (\( k = 1, 2, \ldots, k_1 + k_2 \)); \( R_0 = 0 \).

\( E_{ik} \) Collected amount from site \( i \) in tour \( k \) (\( i \in N, k = 1, 2, \ldots, k_1 + k_2 \)).

\( S_k \) Total collected amount in tour \( k \) (\( k = 1, 2, \ldots, k_1 + k_2 \)).

\( L_{ik} \) Auxiliary variable used to calculate \( E_{ik} \) (\( i \in N, k = 1, 2, \ldots, k_1 + k_2 \)).

\( W_k \) Waiting time of the vehicle at the processing facility at the beginning of tour \( k \) (\( k = 1, 2, \ldots, k_1 + k_2 \)).

\( Q_k \) Amount of unprocessed items at the processing facility at the end of tour \( k \) (\( k = 0, 1, 2, \ldots, k_1 + k_2 \)). \( Q_0 \) is a parameter which represents the number of unprocessed items at the beginning of the day, i.e., \( Q_0 = I_i(0) \).

\( U_k \) Amount of processed items between \( R_k \) and \( R_{k+1} \) (\( k = 0, 1, 2, \ldots, k_1 + k_2 - 1 \)).

\( U_{k_1: k_2} \) Amount of processed items between \( R_{k_1: k_2} \) and \( T_f \).
variables, $L_k$’s, which cannot be larger than $\tau_c$ and $T_k$ through Constraints (5a),(5b). Note that, $L_k$ is set to zero by Constraint (5c). Then, we set $\ell(L_k - L_{k-1})$ as the collected amount from node $i$ in tour $k$ through Constraint (5d). Constraint (5e) ensures that all of the items accumulated at the nodes are collected. Finally, Constraint (5f) calculates the collected amount in each tour.

The amount of processed items between the return times of two consecutive tours, $U_k$, cannot exceed the amount of unprocessed items, $Q_k$, and processing capacity, $\mu(R_0 - R_k)$. This condition is satisfied via Constraints (6a)–(6c). Constraint (7) balances the queue size at the processing facility. Constraint (8) defines the binary variables in a tour, while Constraint (9) dictates nonnegativity of all variables.

Our computational experiments showed that both $M_1$ and $M_2$ can be solved to optimality only for problems with low workload levels (e.g., $x \leq 0.8$) and limited number of nodes (e.g., $n \leq 10$), but verifying optimality takes extensive computation time. In particular, $M_2$ is significantly more difficult than $M_1$. To strengthen model $M_2$, we generate a valid inequality using a lower bound for the daily processed amount, $P^*$. In order to process at least $P^*$ items, the vehicle should complete the first tour by time $\tau_j - P^*/\mu$. That is, $R_0 \leq \tau_j - P^*/\mu$. As the vehicle transports $S_i$ items to the processing facility at the end of tour 1, the processor can process at most $S_1$ items up to $R_2$. Therefore, the second tour cannot be completed later than $\tau_j - (P^* - S_1)/\mu$ so that the remaining $P^* - S_1$ items can be processed in $(P^* - S_1)/\mu$ time units. That is, $R_0 \leq \tau_j - P^*/\mu + S_1/\mu$. Continuing like this, we obtain the following valid inequalities.

Proposition 4.1. The following inequalities are valid for model $M_2$:

$$R_k \leq \tau_j - P^*/\mu + \sum_{p=1}^{k-1} S_p/\mu, \quad \forall k = 1, 2, \ldots, k_1 + k_2.$$  

(Proof is provided in Appendix D.3, which is available as part of the Online Supplement.)

Proposition 4.1 is used in relaxations of model $M_2$ to generate lower bounds. In addition, we tested the inequalities on a set of instances provided in Section 7 and observed that it provides a slight improvement (of around 1–2%) on the best found objective value of $M_2$ at the end of a four-hour (i.e., 14,400 seconds) run time limit. Hence, we included these inequalities in all of our computational tests.

5. Lower bounds on daily transportation cost

In this section, we bound the daily transportation cost, $C$, under the condition that at least $P^*$ items should be processed by time $\tau_j$. We present two relaxations of model $M_2$. The corresponding MIP formulations are solved with a run time limit and the best found lower bound is kept. The highest bound is used to evaluate the performance of our solution approach in terms of daily transportation cost.

In model $M_2$, having Constraint (10) narrows the set of feasible solutions dramatically. Especially when the workload level is high, there may be few solutions that achieve $P^*$. Thus, the branch and bound algorithm has difficulty in finding a candidate feasible solution. To overcome this, we relax Constraint (10) and replace it by the valid inequalities given in Constraint (11). As a result, the relaxed model cannot guarantee that $P^*$ items are processed by time $\tau_j$. Another reason for the difficulty of solving $M_2$ is due to the big-M constraints, i.e., Constraints (4a) and (4b). We eliminate these constraints, which calculate the exact values of $T_k$ variables and prevent subtours. Instead, we utilize a single commodity flow formulation and add the following constraints to the MIP:

$$T_k \leq U_k - \tau_c, \quad \forall i \in N, k = 1, 2, \ldots, k_1 + k_2.$$  

(12)
6.1. Searching for solutions with a final TSP tour

To cope with the computational difficulty of the proposed models, we first restrict the vehicle to perform only one tour that visits all nodes after \( \tau_e \) with duration \( \tau \) and cost \( \theta \). Since items do not accumulate after \( \tau_e \), the node sequence of the final tour does not affect the collected amount, and so, it is independent of the accumulated amounts at sites. Therefore, routing and scheduling decisions before and after time \( \tau_e \) become independent. Hence, we first solve a TSP with the objective of minimizing the total traveling time (not distances) to visit all nodes. This constitutes the last tour. We exclude the visiting variables of the last tour from models \( M_1 \) and \( M_2 \), but in order to calculate the collected and processed amounts we still keep the remaining variables corresponding to the last tour. We generate models \( M_1^T \) and \( M_2^T \) from \( M_1 \) and \( M_2 \), correspondingly, according to the following.

- \( k_2 = 1 \).
- Constraints (2a)–(2d), (3a), (3b), (4a), (4b), and (8) are defined for \( k = 1, 2, \ldots, k_1 \), rather than \( k = 1, 2, \ldots, k_1 + k_2 \).
- Constraints (3c)–(3e) are replaced with
  \[
  \begin{align*}
  R_{s_k+1} & \geq \tau_e + \tau, \\
  R_{s_k} & \leq \tau_e, 
  \end{align*}
  \]
  where Constraint (14a) guarantees that the start time of the final TSP tour is greater than or equal to \( \tau_e \) and Constraint (14b) ensures that at most \( k_1 \) tours are completed before \( \tau_e \).
- Variables \( L_k \), \( \forall i \in N, k = 1, 2, \ldots, k_1 + 1 \), are removed so that Constraints (5a)–(5b) are deleted and Constraints (5c) and (5d) are replaced with
  \[
  \begin{align*}
  T_0 & = 0, \quad \forall i \in N, \\
  E_k & \leq \lambda_i(T_k - T_{i+1}), \quad \forall i \in N, k = 1, 2, \ldots, k_1, \\
  E_{s_k+1} & \leq \lambda_i(\tau_e - T_{i+1}), \quad \forall i \in N, 
  \end{align*}
  \]
  where Constraints (15b) and (15c) do not calculate the exact value of the collected amount from a visited node, but guarantees that the total amount collected from a node by its visit time cannot exceed the accumulated amount after its previous visit.

After solving \( M_2^T \), the cost of the final TSP tour, \( \theta \), is added to the traveling cost.

6.2. Node filtering heuristic

In this section, we propose a node filtering heuristic by adding constraints to models \( M_1 \) and \( M_2 \). These constraints ensure that nodes with low accumulation rates are not allowed to be visited in each tour. By filtering a node, we defer collecting its accumulated amount to a subsequent tour. In order to identify the nodes to be filtered in a tour, we consider the remaining processing capacity and workload. We adjust the remaining processing capacity by assuming that the deferred amount will be processed by \( \tau_f \). If this remaining capacity is sufficient to achieve the upper bound on \( P \), we allow filtering this node.

Specifically, to decide on which nodes to filter in tour \( k \), we consider the following quantities. At the return time of tour \( k - 1 \) (i.e., \( R_{k-1} \)), the total amount of items processed is \( \sum_{i=1}^{k-1} U_i \). Therefore, \( U_B - \sum_{i=1}^{k-1} U_i \) is an upper bound on the processed amount in the time interval \( [R_{k-1}, \tau_k] \), for any \( k \leq k_1 \). Moreover, the remaining processing capacity is \( (\tau_f - R_{k-1})\mu \), while the total accumulation amount in site \( i \) is \( i \tau_e \). Then, the adjusted remaining capacity is \( (\tau_f - R_{k-1})\mu - i \tau_e \). If the following inequality holds
  \[
  (\tau_f - R_{k-1})\mu - i \tau_e \geq UB_P - \sum_{i=1}^{k-2} U_i, 
  \]
  then we still have enough capacity to attain the upper bound even if site \( i \) is not visited in tour \( k \). Hence, site \( i \) can be filtered out in tour \( k \).

Recall that the sites are indexed in non-decreasing order of the accumulation rate. We can generalize the above inequality to a Node Filtering Rule in order to find a threshold value, \( m^* \), such that the first \( m^* \) nodes are filtered out in tour \( k \).

**Definition 2 (Node Filtering Rule).** The vehicle is not allowed to visit sites \( i_1, i_2, \ldots, i_m^* \) in tour \( k \), where \( m^* \) is the largest index such that
  \[
  (\tau_f - R_{k-1})\mu - \sum_{i=1}^{m^*} \lambda_i \geq UB_P - \sum_{i=1}^{k-2} U_i. 
  \]

We incorporate the node filtering approach to models \( M_1 \) and \( M_2 \) and obtain models \( M_1^F \) and \( M_2^F \) by modifying them as follows. We define \( M_2^F = \sum_{i=1}^{m^*} \lambda_i \) and a new decision variable, \( Z_{mk} \).

Next, we add two additional constraints to the formulation:

\[
Y_k \leq 1 - Z_{mk}, 
\quad \forall i \in N, \quad 1 \leq m \leq n, \quad i \in m, \quad k = 1, 2, \ldots, k_1. 
\]

**Constraint (18a)** guarantees that if \( Z_{mk} = 1 \), then a site \( i \), where \( i \in m \), is filtered out in tour \( k = 1, 2, \ldots, k_1 \). Constraint (18b) determines the value of \( m^* \) for tour \( k = 1, 2, \ldots, k_1 \) as defined in the Node Filtering Rule.

For the illustrative example, the node filtering approach generates an optimal solution of \( 0/74.7-4-3-0(142.1)-4-3(221.2)3-4-1-2-0, \) with \( P = 1737 \) and \( C = 182 \). According to the Constraints (18a) and (18b), node 1 is filtered out in the tours before \( \tau_e \). Although node 2 is not filtered out, the models decide not to visit node 2 before \( \tau_e \).

7. Computational experiments

In this section, we analyze the characteristics and performance of the solutions obtained by the proposed solution approach using numerical experiments with realistic data instances. This section starts with a description of our CPP instances and the performance metrics. General results on the performance of our solution approach and detailed analysis on solution characteristics follow. Finally, managerial insights are provided.

7.1. Data instances and performance metrics

Using a real-life data set that we have obtained from a clinical testing laboratory that we have been collaborating with, we constructed 12 problem instance sets. Each instance set is defined on a different complete graph, \( G = (N, E) \), with nodes that correspond to the clients of the clinical laboratory served by a single vehicle in a specified region. The number of nodes in \( N \) varies between 10 and 18. The accumulation rate \( \lambda_i \) for each node \( i \in N \) is determined according to empirical data such that the majority of the volume comes from a small portion of the clients. The distance \( d_{ij} \) for each arc \((i,j) \in E \) is the shortest path distance on the roads obtained from Google Maps. The traveling time, \( t_{ij} \), for each arc \((i,j) \in E \) is taken as \( t_{ij} = d_{ij}/v + s_j \), where \( v \) denotes the constant speed of the vehicle (miles per min) and \( s_j \) denotes the service time at node \( j \) (in minutes). The service time, \( s_j \), represents the amount of time that the driver spends at node \( j \) to pick up the accumulated
items. We randomly generated the \(s_j\) values from a uniform distribution defined on the interval \([2,5]\) if \(s_j \leq 20\) (items per hour), and on the interval \([6,9]\) otherwise, based on empirical data.

For each problem instance set, we generated three workload levels (\(x = 0.8, 1.0, 1.2\)) by varying \(\mu\). The value of \(\mu\) for the corresponding instance is defined using the equality \(\tau_c \sum_{j \in \mathcal{J}} s_j = 3\mu t_j\). As a result, there are 36 data instances in total.

Typically, the sites open at 8 a.m. and close at 7 p.m. Processing at the facility generally ends at 5 a.m. Thus, \(\tau_c\) is set to 660 minutes, and \(\tau_t\) is 1260 minutes, considering minutes as the time unit. The initial queue level at the processing facility is set to zero for each instance. The data sets can be accessed at http://home.ku.edu.tr/eyucel/Research/DatasetCPP.zip.

The computational experiments were performed on a Xeon E5520 @ 2.27 gigahertz processing unit with 48 gigabytes of memory. GAMS 23.3 was used to create the models. CPLEX 12.2 was employed to solve them with the following options turned on: threads 0; parallel mode 1. This setting enables CPLEX to run as a multi-threaded application and to distribute the computing load to as many as eight logical cores of the Core i7 Quad-core processor. The run time limit for each instance was set to 60 minutes, and \(\tau_t\) is 1260 minutes, considering minutes as the time unit. The initial queue level at the processing facility is set to zero for each instance. The data sets can be accessed at http://home.ku.edu.tr/eyucel/Research/DatasetCPP.zip.

The experimental results are reported in Table B.1 in Appendix B and the aggregate results for all instances are provided in Table 3.

We see from Table 3 that the gap between the daily processed amounts of the best found solution by the TF heuristic and the infinite vehicle relaxation solution is 1.07% on the average, while the maximum gap is 3.59%. This shows that this solution approach is very effective in terms of maximizing the daily processed amount. It also indicates that the infinite vehicle relaxation provides tight upper bounds on the test instances. The optimality gap for the daily transportation cost, Gap\(_C\), is 18.06% on the average, while the maximum gap is 29.52%. These relatively large gaps may be attributed to several factors: (1) the weakness of the lower bounds, (2) the possibility that the best found solutions may be far from the optimal, (3) inherent computational difficulty of the problem. Similar to researchers studying challenging routing problems such as

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>5375</td>
<td>183</td>
<td>5375</td>
<td>212</td>
<td>5375</td>
<td>227</td>
<td>5375</td>
<td>213</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5188</td>
<td>356</td>
<td>5188</td>
<td>384</td>
<td>5187</td>
<td>410</td>
<td>5186</td>
<td>381</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>4342</td>
<td>418</td>
<td>4342</td>
<td>334</td>
<td>4342</td>
<td>350</td>
<td>4338</td>
<td>335</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Table 2**

The best found solution values of the models averaged over the 12 instance sets.

The experimental results are reported in Table B.1 in Appendix B and the aggregate results for all instances are provided in Table 3.

| \(P\) | Daily processed amount of the best found solution by the TF heuristic |
| \(C\) | Daily transportation cost of the best found solution by the TF heuristic |
| \(K\) | Number of tours performed up to time \(\tau_t\) for the best found solution |
| \(\nu\) | Percentage of sites that are visited, averaged over \(K\) tours for the best found solution (i.e., \(100\sum_{k=1}^{K} P_k/n\), where \(P_k\) is the number of sites visited in tour \(k\)) |
| \(f\) | Percentage of sites that are filtered, averaged over \(K\) tours for the best found solution (i.e., \(100\sum_{k=1}^{K} f_k/n\), where the first \(f_k\) sites are filtered out in tour \(k\)) |
| \(DA\) | Daily achievement percentage, calculated as \(100P/(\tau_c \sum_{j \in \mathcal{J}} s_j)\) |
| \(DU\) | Daily capacity utilization percentage, calculated as \(100P/(\mu \tau_t)\) |
| \(Gap_P\) | Percentage gap between \(P\) and UB\(_P\), calculated as \(100(UB_P - P)/P\) |
| \(Gap_C\) | Percentage gap between \(C\) and LB\(_C\), where LB\(_C\) is the best bound found by CPLEX for the two relaxations at the end of the given run time limit (i.e., \(100(C - LB_C)/LB_C\))

## 7.2. Experimental results

In this section, we first compare the solutions found by the proposed models in terms of the daily processed amount and the daily transportation cost for a set of instances with different workload levels. Next, we report the solutions of the best performing model with respect to the deviations from the upper bound presented in Section 3 on the daily processed amount and the best lower bound obtained from the two relaxations given in Section 5 on the daily transportation cost. In addition, we investigate the effect of the workload level and different accumulation rate patterns on the solutions and the problem difficulty. Finally, we analyze the properties of the best found solutions to derive some managerial insights.

### 7.2.1. Comparison of the proposed models

By determining the appropriate sites to be filtered out in each tour, the node filtering approach and the assumption of a final TSP tour narrow the search space, and thus, improve the best found solution by the branch-and-bound process within the given time limit. In order to demonstrate its effectiveness, for all of the 36 instances we solve the models that seek four different types of solutions: (i) with a final TSP tour and the node filtering constraints (referred to as \(M^1_i\) and \(M^2_i\)), (ii) without a final TSP tour and with node filtering constraints (\(M^3_i\) and \(M^4_i\)), (iii) with a final TSP tour and without node filtering constraints (\(M^1_i\) and \(M^2_i\)), and (iv) without a final TSP tour and node filtering constraints (\(M^1_i\) and \(M^2_i\)). Table 2 provides the best found solution values at the end of a given computation time limit (two hours for model \(M^1\)) and four hours for model \(M^2\) averaged among all instances corresponding to each workload level. We note that Table 2 reports the results of \(M^1_i\), \(M^2_i\), \(M^3_i\), and \(M^4_i\) with \(k_1 = 10\), and \(M^1_i\), \(M^2_i\), \(M_i\), and \(M^1_i\) with \((k_1, k_2) = (10, 3)\) as these values perform the best in our experiments with different settings of \(k_1\) and \(k_2\).

Table 2 shows that all models are equally effective in finding the maximal daily processed amount, \(P\). However, in terms of the daily transportation cost, \(C\), \(M^1_i\) clearly performs the best. Based on these observations demonstrating the effectiveness of the node filtering approach with a final TSP tour, the remaining experiments are performed using models \(M^1_i\) and \(M^2_i\). We refer to the heuristic approach based on solving these two models within the given time limit as the "TF heuristic hereafter."

### 7.2.2. Test results and performance of the TF heuristic

We now evaluate the performance of solutions obtained by the TF heuristic using the upper bound on the daily processed amount generated by the procedure in Section 3 and the lower bound on the daily transportation cost generated from the MIP relaxations in Section 5 with \(k_1 = 10\). For each instance, we report the following results.

| \(P\) | Daily processed amount of the best found solution by the TF heuristic |
| \(C\) | Daily transportation cost of the best found solution by the TF heuristic |
| \(K\) | Number of tours performed up to time \(\tau_t\) for the best found solution |
| \(\nu\) | Percentage of sites that are visited, averaged over \(K\) tours for the best found solution (i.e., \(100\sum_{k=1}^{K} P_k/n\), where \(P_k\) is the number of sites visited in tour \(k\)) |
| \(f\) | Percentage of sites that are filtered, averaged over \(K\) tours for the best found solution (i.e., \(100\sum_{k=1}^{K} f_k/n\), where the first \(f_k\) sites are filtered out in tour \(k\)) |
| \(DA\) | Daily achievement percentage, calculated as \(100P/(\tau_c \sum_{j \in \mathcal{J}} s_j)\) |
| \(DU\) | Daily capacity utilization percentage, calculated as \(100P/(\mu \tau_t)\) |
| \(Gap_P\) | Percentage gap between \(P\) and UB\(_P\), calculated as \(100(UB_P - P)/P\) |
| \(Gap_C\) | Percentage gap between \(C\) and LB\(_C\), where LB\(_C\) is the best bound found by CPLEX for the two relaxations at the end of the given run time limit (i.e., \(100(C - LB_C)/LB_C\))

### Table 3

Aggregate test results.

<table>
<thead>
<tr>
<th>(K)</th>
<th>(\nu (%))</th>
<th>(f (%))</th>
<th>(DA (%))</th>
<th>(DU (%))</th>
<th>(Gap_P (%))</th>
<th>(Gap_C (%))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg.</td>
<td>6.78</td>
<td>14.9</td>
<td>53.9</td>
<td>91.51</td>
<td>91.58</td>
<td>1.07</td>
</tr>
<tr>
<td>Min.</td>
<td>3</td>
<td>7.14</td>
<td>28.79</td>
<td>69.63</td>
<td>80.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Max.</td>
<td>10</td>
<td>25</td>
<td>80.95</td>
<td>100.00</td>
<td>97.48</td>
<td>3.59</td>
</tr>
</tbody>
</table>
IRP and TOP (Moin and Salhi, 2007; Archetti et al., 2007), we have observed that finding a good lower bound for CPP is very difficult. Although our problem relaxations given in Section 5 provide substantial improvement in achieving stronger bounds, there is still room for further improvement.

### 7.2.3. Effect of workload level

In order to investigate the effect of the workload level on the solutions and the problem difficulty, we aggregate in Table 4 the results by workload level and provide averaged values corresponding to the same workload level. We see that the vehicle performs less number of tours when \( \alpha = 0.8 \), since a significant portion of the total accumulation can be processed after the final TSP tour. When \( \alpha > 1.0 \), the vehicle performs more tours to prevent idleness of the processing unit during the time interval \((0, \tau_c)\). Correspondingly, the daily transportation cost becomes higher. When \( \alpha \) increases to 1.2, the vehicle starts to perform less number of tours compared to the \( \alpha = 1 \) case, since it collects a smaller volume up to time \( \tau_c \) as processing the total accumulation is not possible. In the best found solutions, independent of the workload level, the vehicle visits about 15% of the sites in each tour on the average, implying that it is generally sufficient to visit only a small subset of the nodes to collect “enough” items. In addition, at least 50% of the sites are filtered out on the average in each tour without much loss from optimality. For small \( \alpha \), more nodes can be filtered out. For all instances with \( \alpha = 0.8 \), we see that 80% of daily processing capacity utilization is sufficient to process all of the daily accumulated amount. For the instances with \( \alpha \geq 1 \), although all of the daily accumulation cannot be processed, 96.72% of the daily processing capacity is utilized on the average. As expected, the average DA values in Table 4 show that the percentage of items processed on time decreases as the workload increases. On the other hand, the processing capacity utilization increases in the workload level. The gaps reported for the best found solutions in terms of both the daily processed amount and the daily transportation cost are the highest for \( \alpha = 1.0 \), which generally represents a difficult case.

### 7.2.4. Effect of accumulation rate pattern

The accumulation rate pattern of the sites is an influential problem characteristic in CPP since nodes are selectively visited during the tours before \( \tau_c \). To evaluate the performance of the TF heuristic in settings with different accumulation rate patterns, we extend our tests to new data sets. We pick the first instance set in Table B.1, and in addition to the original pattern (denoted by \( O \)), we generate new instance sets by distributing the total daily accumulation randomly in four patterns: (i) Pareto Random (\( PR \)), where 20% of sites contribute to 80% of the whole accumulation; (ii) Pareto Closer (\( PC \)), where 20% of sites that are closest to the processing facility accumulate 80% of the whole accumulation; (iii) Pareto Distant (\( PD \)), where the closest 80 percent of sites contribute to only 20% of the whole accumulation; and (iv) Identical (\( I \)), where all accumulation rates are identical. In all of the patterns, the sum of the accumulation rates is equal to that of the original one. For the selected problem instance, the geographical locations of the sites (nodes 1–14) and the processing facility (node 0) are illustrated in Fig. E1 in Appendix E, which is available as part of the Online Supplement. The accumulation rates, the traveling times to processing facility, and the service times for each site for pattern \( O \) are also provided in Table E2 in Appendix E. In Fig. E1, the sites are colored such that darker ones have a higher accumulation rate.

The experimental results reported in Table 5 indicate that the performance of the TF heuristic, measured by average Gap\(_P\) and Gap\(_C\), decreases in the order of \( PC, PR, PD, I \). Since the accumulation rates are identical for all sites in pattern \( I \), the filtering rule can filter out only a small percentage of sites (around 10%); thus, the accelerating effect of the node filtering approach on the branch-and-bound algorithm deteriorates. In \( PC \), which resembles the original pattern observed at the clinical laboratory we have been collaborating with, our approach shows the best performance by filtering nearly 50% of the sites. Although the percentage of filtered sites are also nearly 50% in \( PD \), the solution quality gets worse especially in traveling cost (Gap\(_C = 30\% \)). While the filtered node percentage is only 34% in pattern \( PR \), our solution approach still provides comparable average gaps (1.3% for the daily processed amount and 28.53% for the daily transportation cost).

In conclusion, we observe that the filtering approach is effective in the Pareto cases, which are probably the most realistic. The extreme case with identical accumulation rates, which is unlikely to be seen in practice, results in relatively poor performance.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>( \alpha )</th>
<th>( P )</th>
<th>( C )</th>
<th>( K )</th>
<th>( f ) (%)</th>
<th>DA (%)</th>
<th>DU (%)</th>
<th>Gap(_P) (%)</th>
<th>Gap(_C) (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>0.8</td>
<td>10,493</td>
<td>154</td>
<td>3</td>
<td>9.52</td>
<td>59.67</td>
<td>100.00</td>
<td>80.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>10,315</td>
<td>308</td>
<td>10</td>
<td>25</td>
<td>53.00</td>
<td>98.30</td>
<td>98.30</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>8612</td>
<td>249</td>
<td>9</td>
<td>22.2</td>
<td>39.51</td>
<td>82.07</td>
<td>98.48</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td>9807</td>
<td>237</td>
<td>7.33</td>
<td>18.92</td>
<td>50.72</td>
<td>93.46</td>
<td>92.26</td>
<td>0.50</td>
</tr>
<tr>
<td>PR</td>
<td>0.8</td>
<td>10,493</td>
<td>169</td>
<td>3</td>
<td>16.7</td>
<td>28.57</td>
<td>100.00</td>
<td>80.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>10,192</td>
<td>381</td>
<td>8</td>
<td>31.3</td>
<td>50.00</td>
<td>97.13</td>
<td>97.13</td>
<td>1.99</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>8511</td>
<td>305</td>
<td>9</td>
<td>38.1</td>
<td>25.40</td>
<td>81.11</td>
<td>97.33</td>
<td>1.90</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td>9732</td>
<td>285</td>
<td>6.67</td>
<td>28.67</td>
<td>34.66</td>
<td>92.75</td>
<td>91.49</td>
<td>1.30</td>
</tr>
<tr>
<td>PD</td>
<td>0.8</td>
<td>10,493</td>
<td>223</td>
<td>4</td>
<td>16.1</td>
<td>65.75</td>
<td>100.00</td>
<td>80.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>10,035</td>
<td>419</td>
<td>8</td>
<td>31.3</td>
<td>59.25</td>
<td>95.63</td>
<td>95.63</td>
<td>2.56</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>8391</td>
<td>367</td>
<td>6</td>
<td>39.3</td>
<td>25.07</td>
<td>79.97</td>
<td>95.96</td>
<td>2.31</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td>9640</td>
<td>336</td>
<td>6.00</td>
<td>28.87</td>
<td>50.02</td>
<td>91.87</td>
<td>90.53</td>
<td>1.62</td>
</tr>
<tr>
<td>I</td>
<td>0.8</td>
<td>10,493</td>
<td>303</td>
<td>3</td>
<td>85.7</td>
<td>7.14</td>
<td>100.00</td>
<td>80.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>9891</td>
<td>511</td>
<td>9</td>
<td>23</td>
<td>11.11</td>
<td>94.26</td>
<td>94.26</td>
<td>4.69</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>8323</td>
<td>451</td>
<td>10</td>
<td>26.4</td>
<td>11.43</td>
<td>79.32</td>
<td>95.18</td>
<td>3.78</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td>9569</td>
<td>422</td>
<td>7.33</td>
<td>45.05</td>
<td>9.89</td>
<td>91.19</td>
<td>89.81</td>
<td>2.82</td>
</tr>
</tbody>
</table>
7.2.5. Effect of positive initial workload

As we see in Table B.1, for some instances, all of the daily accumulation amount cannot be processed by time \( \tau_0 \). In those cases, the unprocessed items that remain at time \( \tau_0 \), which is equal to \( \sum_{i \in N} s_i - P \), may constitute the initial workload for the next day. These items can be processed while the vehicle is performing its first tour, and this reduces the idle time.

In this section, we solve a five-day problem for the week days, and analyze the effect of positive initial workload on the daily processed amount, \( P \), for each day. For the first day, we solve the models with \( Q_0 = 0 \). For the following days, we solve the models with \( Q_0 \) set to the unprocessed amount of the previous day, i.e., \( \sum_{i \in N} s_i - P \). We assume that the same accumulation rates are observed every day and items remaining from the previous day are processed first. Table 6 reports \( Q_0 \) and \( P \) values for the best found solutions of each day for the first instance set in Table B.1 with \( x = 0.8, 1, \) and 1.2. According to the results in Table 6, when \( x \geq 1 \), allowing initial backlog increases the utilization of the processing facility. In the case of \( x = 1 \), the system becomes stable after the first day. In the case of \( x = 1.2 \), although the processing capacity is fully utilized after the first day, the initial workload increases day by day. This also illustrates the effect of the workload parameter on the system performance when backlogging is possible.

7.2.6. Managerial insights

All of the test results up to this point show that both the solution quality and the characteristics of the solution change as the workload level changes. In this section, we further investigate the real-life implications of this factor by additional tests with the aim of providing some generalizable insights. We pick, again, the first instance set in Table B.1 and solve it for a number of different \( x \) values varying between 0.8 and 1.4 (obtained by decreasing the processing rate \( \mu \) while keeping the daily accumulated amount constant).

According to the results presented in Table 7, we observe that all of the daily accumulation can be processed when \( x < 1.0 \). As the workload level increases, the percentage of the daily accumulation that can be processed on time decreases, whereas the processing capacity utilization increases. The gap between the daily processed amount of the solution and the corresponding infinite vehicle relaxation solution is the largest when \( x = 1.0 \) and it decreases as the workload exceeds 1.

When \( x < 1.0 \), idle times at the processing unit are tolerable during \((Q_0,\tau_0)\) since most of the items can be processed after the final TSP tour. Since a smaller number of tours is sufficient, the best found daily transportation costs are low. When \( x = 0.8 \), the problem becomes relatively easy as around 81% of nodes can be filtered without causing any significant loss from optimality. As \( x \) increases to 0.9, it is more difficult to identify the nodes to be filtered (\( f = 44.64\% \)). In addition, the percentage of nodes visited in a tour increases from 7.14% to 10.7%, causing the transportation cost to increase by 24%. In case of \( x = 1.0 \), the problem becomes significantly more difficult since the number of tours required almost doubles. The vehicle visits the same percentage of nodes in a tour but has to perform more frequent tours to ensure that the processing center is well-supplied with the items to be processed. The processing rate decreases as \( x \) increases, so that it requires more time to process the same amount of items. As a result, the best found daily transportation cost decreases in \( x \).

When \( x \geq 1.0 \), capacity utilization remains constant due to routing restrictions. For \( x = 1.4 \), both \( M_1 \) and \( M_2 \) can be solved to near-optimality since filtering becomes more effective while both \( K \) and \( P \) decrease.

The best found solutions by the TF heuristic for each workload level are provided in Table 8, where the numbers in parentheses denote the amount of waiting times at the corresponding nodes. The TSP tour performed after time \( \tau_0 \) is 8-6-2-7-14-5-11-13-4-9-3-12-10-1 with a cost of 108. According to these solutions, before the first tour a longer waiting time can be tolerated when \( x < 1 \). Thus, all of the daily accumulation can be processed easily on time. For larger workload levels, the waiting time before the first tour is smaller. For all workload levels, it is better to visit closer sites with high accumulation rates (i.e., nodes 12, 13, and 14) before \( \tau_0 \). In none of the solutions, nodes 1–7, which have the lowest accumulation rates, are visited before \( \tau_p \). For these nodes, the ratio of the accumulation rate to the total time to service a node and return to the processing facility (given in Table E2) is less than 0.01.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( \sum_{i \in N} s_i / \tau_0 )</th>
<th>( \mu )</th>
<th>1st day</th>
<th>2nd day</th>
<th>3rd day</th>
<th>4th day</th>
<th>5th day</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>10,493</td>
<td>13,117</td>
<td>0</td>
<td>10,185</td>
<td>0</td>
<td>10,493</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>10,493</td>
<td>10,493</td>
<td>0</td>
<td>1,015</td>
<td>308</td>
<td>10,493</td>
<td>308</td>
</tr>
<tr>
<td>1.2</td>
<td>10,493</td>
<td>8744</td>
<td>0</td>
<td>8507</td>
<td>1986</td>
<td>8744</td>
<td>3735</td>
</tr>
</tbody>
</table>

Table 6: Best found solution values of the models for five sequential days for the selected instance set.

<table>
<thead>
<tr>
<th>( x )</th>
<th>( \sum_{i \in N} s_i / \tau_0 )</th>
<th>( \mu )</th>
<th>1st day</th>
<th>2nd day</th>
<th>3rd day</th>
<th>4th day</th>
<th>5th day</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>10,493</td>
<td>13,117</td>
<td>0</td>
<td>10,185</td>
<td>0</td>
<td>10,493</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>10,493</td>
<td>10,493</td>
<td>0</td>
<td>1,015</td>
<td>308</td>
<td>10,493</td>
<td>308</td>
</tr>
<tr>
<td>1.2</td>
<td>10,493</td>
<td>8744</td>
<td>0</td>
<td>8507</td>
<td>1986</td>
<td>8744</td>
<td>3735</td>
</tr>
</tbody>
</table>

Table 7: Test results for the selected problem instance for varying workload levels.

<table>
<thead>
<tr>
<th>( x )</th>
<th>Best found solution</th>
<th>Processor idle time (min)</th>
<th>Vehicle idle time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>0(15)-14-0(98)-4(139.16)-15-0(181.71)-12-0(28.1)-14-0-TSP-0</td>
<td>252</td>
<td>506</td>
</tr>
<tr>
<td>0.9</td>
<td>0(100)-14-0(85.35)-12-0(16.43)-14-0(13.9)-0(19.28)-14-0(64.94)-TSP-0</td>
<td>126</td>
<td>465</td>
</tr>
<tr>
<td>1.0</td>
<td>0-13-14-0(1.2)-14-0(1.4)-0(10.18)-14-0(1.39)-14-0(29.04)-14-0-TSP-0</td>
<td>37</td>
<td>294.6</td>
</tr>
<tr>
<td>1.1</td>
<td>0(0.66)-8-9-14-0(12.28)-10-12-0(33.9)-14-0(63.1)-13-14-0(181.35)-14-0-TSP-0</td>
<td>35.7</td>
<td>291.3</td>
</tr>
<tr>
<td>1.2</td>
<td>0(4.15)-9-14-0(14.4)-13-14-0(58.71)-10-12-0(25.16)-14-0(180.81)-10-0(0.03)-14-0-TSP-0</td>
<td>34.2</td>
<td>270.3</td>
</tr>
<tr>
<td>1.3</td>
<td>0(3.8)-9-14-0(14.4)-13-14-0(58.17)-12-0(23.2)-14-0(181)-14-0-TSP-0</td>
<td>32.8</td>
<td>279.8</td>
</tr>
<tr>
<td>1.4</td>
<td>0(1.41)-9-14-0(17.91)-14-0(64.55)-12-0(14.51)-14-0(175.72)-14-0-TSP-0</td>
<td>31.7</td>
<td>274.4</td>
</tr>
</tbody>
</table>

Table 8: The best found solutions for the selected instance for varying workload levels.
In Table 8, we report the idle time of the processing facility throughout the day and the idle time of the vehicle during the working day (i.e., between times 0 and \( \tau_e \)). As the workload level increases, the idle times of the processor decrease, while the vehicle stays almost at the same level of idleness for \( \alpha \geq 1 \).

### 8. Conclusions

We considered the clinical specimen collection problem under the objectives of maximizing the daily processed amount as a first priority and minimizing the daily transportation costs as a second priority. We defined the Collection for Processing Problem (CfPP) as the problem of designing vehicle routes throughout the day to match the collected workload with the processing capacity. We showed through computational tests that our approach can be ruled out without significantly compromising from optimality.

Future work includes developing heuristic algorithms to solve larger CfPP instances. Additional considerations to be incorporated into the formulation include multiple vehicles and stochastic accumulation of specimens at the sites.

---

**Appendix A. Notation**

\( n \) Number of sites

\( \tau_e \) Time of day when sites close

\( \tau_f \) Time of day when processing at the processing facility ends

\( t_{ij} \) Traveling time from node \( i \) to node \( j \), where \( i, j \in N^* \)

\( d_{ij} \) Distance from node \( i \) to node \( j \), where \( i, j \in N^* \)

\( \lambda_i \) Accumulation rate at site \( i \) per unit time, where \( i \in N \)

\( \mu \) Processing rate per unit time of the processing facility

\( k_1 \) Maximum number of tours that can be performed until time \( \tau_e \)

\( k_2 \) Maximum number of tours that can be performed between times \( \tau_e \) and \( \tau_f \)

\( \alpha \) Workload level, the ratio of daily accumulation to daily processing capacity, \( \alpha = \frac{\tau_f - \tau_e}{\tau_f \mu} \)

\( G \) Set of feasible solutions for a CfPP instance

\( g \) A solution in \( G \)

\( P(t) \) Amount of items processed between time 0 and \( t \), where \( t \in [0, \tau_f] \), \( P(t) \in \mathbb{R}^+ \)

\( C(t) \) Transportation cost between time 0 and \( t \), where \( t \in [0, \tau_f] \), \( C(t) \in \mathbb{R}^+ \)

\( UB_p \) Upper bound for the daily processed amount, \( UB_p \in \mathbb{R}^+ \)

\( LB_C \) Lower bound for the daily transportation cost, \( LB_C \in \mathbb{R}^+ \)

\( R_k \) Return time of tour \( k \), where \( k = 1, 2, \ldots, k_1 + k_2 \) and \( R_k \in \mathbb{R}^+ \)

\( W_k \) Amount of waiting time before tour \( k \), where \( k = 1, 2, \ldots, k_1 + k_2 \) and \( W_k \in \mathbb{R}^+ \)

\( I_i(t) \) Amount of items at node \( i \) at time \( t \), where \( i \in N^*, \tau_i \in (0, \tau_f) \) and \( I_i(t) \in \mathbb{R}^+ \)

\( Q_k \) Amount of unprocessed items at the end of tour \( k \), where \( k = 1, 2, \ldots, k_1 + k_2 \) and \( Q_k \in \mathbb{R}^+ \) (\( Q_k = I_k(R_k) \))

\( E_k \) Collected amount from node \( i \) in tour \( k \), where \( i \in N, k = 1, 2, \ldots, k_1 + k_2 \) and \( E_k \in \mathbb{R}^+ \)

\( S_k \) Total collected amount in tour \( k \), where \( k = 1, 2, \ldots, k_1 + k_2 \) and \( S_k \in \mathbb{R}^+ \)

\( U_{k_1+k_2} \) Amount of items processed between tour \( (k_1 + k_2)'s \) end time and tour \( (k + 1)'s \) end time, where \( k = 1, 2, \ldots, k_1 + k_2 - 1 \) and \( U_{k_1+k_2} \in \mathbb{R}^+ \) (\( U_{k_1+k_2} = P(R_{k+1} - R_k) \))

---

**Appendix B. Results**

### Table B.1

<table>
<thead>
<tr>
<th>Instance</th>
<th>( n )</th>
<th>( \alpha )</th>
<th>( P )</th>
<th>( C )</th>
<th>( K )</th>
<th>( f )</th>
<th>( DA ) (%)</th>
<th>( DU ) (%)</th>
<th>Gap __ (%)</th>
<th>Gap __ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14</td>
<td>0.8</td>
<td>10,493</td>
<td>171</td>
<td>4</td>
<td>7.14</td>
<td>80.95</td>
<td>97.00</td>
<td>80.00</td>
<td>6.88</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>1,185</td>
<td>321</td>
<td>9</td>
<td>10.7</td>
<td>59.82</td>
<td>97.06</td>
<td>97.06</td>
<td>12.3</td>
<td>15.05</td>
</tr>
<tr>
<td>1</td>
<td>1.2</td>
<td>8,507</td>
<td>284</td>
<td>8</td>
<td>11.2</td>
<td>63.27</td>
<td>81.07</td>
<td>97.29</td>
<td>1.07</td>
<td>14.06</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>4,786</td>
<td>143</td>
<td>4</td>
<td>14.8</td>
<td>74.07</td>
<td>97.00</td>
<td>80.00</td>
<td>0.00</td>
<td>6.72</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
<td>4,658</td>
<td>328</td>
<td>9</td>
<td>23.6</td>
<td>68.06</td>
<td>97.33</td>
<td>97.33</td>
<td>1.21</td>
<td>25.67</td>
</tr>
<tr>
<td>2</td>
<td>1.2</td>
<td>3,886</td>
<td>260</td>
<td>9</td>
<td>16.7</td>
<td>43.06</td>
<td>81.21</td>
<td>97.45</td>
<td>27.45</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>4,522</td>
<td>173</td>
<td>4</td>
<td>11.1</td>
<td>33.33</td>
<td>98.00</td>
<td>98.00</td>
<td>0.00</td>
<td>8.13</td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>4,389</td>
<td>352</td>
<td>10</td>
<td>13</td>
<td>48.15</td>
<td>97.05</td>
<td>97.05</td>
<td>1.24</td>
<td>27.88</td>
</tr>
<tr>
<td>3</td>
<td>1.2</td>
<td>3,665</td>
<td>306</td>
<td>8</td>
<td>13.1</td>
<td>34.52</td>
<td>81.04</td>
<td>97.25</td>
<td>11.3</td>
<td>18.15</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
<td>4,509</td>
<td>218</td>
<td>3</td>
<td>23.1</td>
<td>42.31</td>
<td>100.00</td>
<td>98.00</td>
<td>2.00</td>
<td>7.92</td>
</tr>
<tr>
<td>4</td>
<td>1.0</td>
<td>4,361</td>
<td>415</td>
<td>10</td>
<td>13.7</td>
<td>51.28</td>
<td>96.71</td>
<td>96.71</td>
<td>1.56</td>
<td>28.88</td>
</tr>
<tr>
<td>4</td>
<td>1.2</td>
<td>3,651</td>
<td>361</td>
<td>8</td>
<td>15.4</td>
<td>53.85</td>
<td>98.97</td>
<td>97.16</td>
<td>26.13</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14</td>
<td>4,540</td>
<td>215</td>
<td>3</td>
<td>17.9</td>
<td>42.86</td>
<td>100.00</td>
<td>98.00</td>
<td>1.10</td>
<td>11.40</td>
</tr>
<tr>
<td>5</td>
<td>1.0</td>
<td>4,362</td>
<td>379</td>
<td>8</td>
<td>16.3</td>
<td>37.76</td>
<td>96.07</td>
<td>96.07</td>
<td>2.30</td>
<td>27.61</td>
</tr>
<tr>
<td>5</td>
<td>1.2</td>
<td>3,668</td>
<td>351</td>
<td>8</td>
<td>13.3</td>
<td>56.12</td>
<td>80.80</td>
<td>96.96</td>
<td>1.38</td>
<td>29.52</td>
</tr>
<tr>
<td>6</td>
<td>11</td>
<td>4,928</td>
<td>169</td>
<td>3</td>
<td>18.2</td>
<td>72.73</td>
<td>100.00</td>
<td>98.00</td>
<td>2.00</td>
<td>4.97</td>
</tr>
<tr>
<td>6</td>
<td>1.0</td>
<td>4,746</td>
<td>334</td>
<td>7</td>
<td>21.2</td>
<td>50.00</td>
<td>96.30</td>
<td>96.30</td>
<td>1.93</td>
<td>27.97</td>
</tr>
<tr>
<td>6</td>
<td>1.2</td>
<td>3,983</td>
<td>329</td>
<td>8</td>
<td>14.3</td>
<td>71.43</td>
<td>80.82</td>
<td>96.98</td>
<td>1.23</td>
<td>28.00</td>
</tr>
<tr>
<td>7</td>
<td>11</td>
<td>4,807</td>
<td>170</td>
<td>3</td>
<td>22.7</td>
<td>59.09</td>
<td>100.00</td>
<td>98.00</td>
<td>0.00</td>
<td>6.92</td>
</tr>
</tbody>
</table>

(continued on next page)
References


Appendix C. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.ejor.2012.10.044.