A Two-Campus Transport Problem

Elad Shufan · Hagai Ilani · Tal Grinshpoun

Abstract The presented two-campus transport problem (TCTP) is a pickup-and-delivery scheduling problem. It is a problem of a college with two campuses located at two different cities. Lecturers living in one city should sometimes teach at the other city campus. The problem is that of transporting $N$ lecturers from one campus to the other, using a fleet of $k$ vehicles, so as to minimize the time the lecturers wait for their transport. This is an optimal set partition problem. We mathematically model the one-directional case of the TCTP. The feasible space solution is represented by linear extensions of the requested departure time partially ordered set and reduction to the shortest path problem. This enables finding the optimal solution for the one directional case. TCTP is a two-destinations version of a dial-a-ride problem (DARP).

1 Introduction

Problems of transport are of major concern to many organizations. Examples can be found in distribution centers transporting merchandise to their customers, in institutes which employ a large number of workers, taking care for their transportation, etc. Many variations of transport problems are studied in real life and in the academic literature. These are commonly NP-Hard. Heuristic and Metaheuristic methods where developed for solving such problems [2,3,9,11,14].

Here we present a unique transport problem we got familiar with at our college. The college has two campuses located at two southern Israeli cities, "Ashdod" and "Beer Sheva", denoted here by "A" and "B" respectively. The college logistics should manage...
the transportation of lecturers living in city "A" to campus "B" and vise-versa. Few weeks before the semester starts, each lecturer, knowing her teaching hours, requests for the time she wants to leave her hometown (either "A" or "B") and the time she wants to go back home from the distanced campus ("B" or "A"). The college has a finite number of vehicles of different capacities (usually between 5-9 passengers), and an optimal set of transports is needed to be found. Optimal here refers to the waiting time of the lecturers. Although motivated by a real life problem, the focus of the paper is theoretical.

The two-campus transport problem (TCTP) can be viewed as a version of the well known dial-a-ride problem (DARP) [6,9]. In a DARP instance, a set of ride-requests is given, each ride has a source and a destination. The rides should be operated by a fleet of vehicles in order to optimize some objective function. The complexity of different versions of DARP is discussed by Paepe et. al. [10]. TCTP is a special case of a DARP, where all the sources and destinations are either "A" or "B". We also distinguish between the two different types of ride-requests (or passengers), which we denote "s-type" and "r-type". "S-type" passengers are being transported to teach, i.e., they are just starting their daily teaching obligation, while passengers of "r-type" have already finished their lecture and are returning back home. Some DARP's also distinguish between "inbound" and "outbound" requests, correspond to "r" and "s" types respectively. The distinction is expressed by a time-window which accompanies the request. A lecturer who has finished teaching at $t_r$, determines the ride's release time, and a largest possible time window is $[t_r, t_{fin}]$, where $t_{fin}$ is the last possible time for the transport departure, determined by the end of the day. A lecturer who is going to teach determines a deadline time $t_d$, and the corresponding largest time window is $[t_{initial}, t_d]$, where $t_{initial}$ is the beginning-of-the-day time. The most relevant objective function in the DARP literature is the latency, determined by the sum of the completion time of all the rides (denoted $\sum C_i$ in [10]). As for this latency function, "the sooner the better", the request should be fulfilled as early as possible. However, the TCTP is different because when an "s-type" request is considered, the lecturer usually wants the request to be executed as close as possible to the deadline, definitely not as soon as possible. The TCTP objective we take treats the dissatisfaction of customers with regard to the aberration between the real pick-up time and the requested time, distinguishing between s-type and r-type requests. This is usually not considered in a DARP. The TCTP is therefore a special case of a DARP which is not considered in [10]. Besides a simple one directional case of TCTP (section 3), which is indeed simple, the complexity of the TCTP is yet not known.

Most of the article treats the one-directional case, namely the arrangement of $N$ passengers into $k$ groups, each group represents a transport going from one city to the other, without taking into account the fact that the same vehicles are needed to take passengers also on the other direction. In other words, we do not check the availability of the transport. In what follows, when discussing the one-directional case, we treat the transportation of $N$ passengers from "A" to "B", assuming that $k$ vehicles all wait at "A" for their departure.

We state three main levels of problem complexity, with increasing difficulty:

1. The one-directional case, with passengers of one type, i.e, either all the passengers are going to teach, or all passengers have just finished lecturing and want to go back to their hometown. We mathematically define this relatively simple problem
and, for a fleet of uniform capacity, suggest a complete polynomial time solution. The solution is based on a reduction of an ordered set optimal partition problem [4] to the known shortest path problem [1,8].

2. The one-directional case, with passengers of two types. We present the problem mathematically and suggest an algorithm for obtaining the complete space solution. This is done by generation of the linear extensions of a partially ordered set and a reduction to the shortest path problem. This provides a practical solution to a relatively small solution space and uniform capacity fleet.

3. The general case of two directions, with two types of passengers, and a fixed number of vehicles of known capacities.

Note that each of the three sub-problems can have some variations – passengers might have a different level of importance, the two-directional problem can have only one passenger type, etc.

After giving some basic notations and principles on section 2, we describe the simple one-directional case (section 3) and the complex one (section 4). On the final section we conclude.

2 Definitions and problem statement

Two sets are given \( S = \{s_1, s_2, \ldots, s_m\} \) and \( R = \{r_{m+1}, r_{m+2}, \ldots, r_{m+n}\} \), representing the requested departure times of s-type and r-type lecturers, respectively. The target is to divide \( S \cup R \) into \( k \) subsets, each represents one transport, and to define the departure time of each transport, so as to minimize the waisted time lectures wait for their transport. A waisted time of one lecturer is defined as the difference between the requested time and the actual departure time of the transport the lecturer was placed at. Note that we actually divide the set \( \{1, 2, \ldots, N\} \), with \( N = m + n \).

2.1 Objective function

The college logistics are expected to minimize the number of transports, as well as the waisted time. These two objectives are in conflict – if we increase the number of transports, we decrease the waisted time (obviously when the number of transports is equal to the number of lecturers, no one waits, but the cost is high). Our aim is to find the best transport arrangement, for a given number of transports, \( k \). The transport manager should start with the minimal \( k \), compute the best arrangement, then increase \( k \), check the improvement and decide if it justifies adding a transport. Iteratively, she goes on increasing \( k \), until an optimal balance between time and money is achieved.

We consider two common objectives of minimizing the waisted time. The first, the additive approach, is to minimize the sum of waisted times of all the lecturers. The second, the min-max approach, is to minimize the maximal waisted time. Let \( \{i_1, i_2, \ldots, i_a, i_{a+1}, i_{a+2}, \ldots, i_b\} \), with \( b \geq a \), be a subset representing the requested times \( r_{i_1} \leq r_{i_2} \leq \cdots \leq r_{i_a} \leq s_{i_{a+1}} \leq s_{i_{a+2}} \cdots \leq s_{i_b} \), of lecturers sharing the same transport. Inequality \( r_{i_a} \leq s_{i_{a+1}} \) is a necessary condition – an r-type lecturer, who wants to return after finishing her lecture, cannot share a transport with an s-type lecturer who needs to depart earlier. Denote the departure time by \( t \). The contribution of the given transport to the additive objective function \( f_{\text{Additive}} \) is given by
\[ \sum_{j=1}^{a} (t - r_{ij}) + \sum_{j=a+1}^{b} (s_{ij} - t) \]

Considering the min-max objective function, \( f_{\text{MinMax}} \), the maximal waisted time of this transport is

\[ \max\{ (t - r_{i1}), (s_{i_b} - t) \} \]

The optimal departure time of the given transport \( t_{OPT} \) is determined in the following way: if \( b - a = 0 \) (i.e. no starting lecturers), then \( t_{OPT} = r_{ia} \). If \( a = 0 \) (no returning lecturers), \( t_{OPT} = s_{ia+1} \). Otherwise (\( 0 < a < b \)), \( t_{OPT} \) depends on the objective function. When the additive objective function is considered

\[
t_{OPT} = \begin{cases} 
  r_{ia} & a > b - a \\
  (r_{ia} + s_{ia+1})/2 & a = b - a \\
  s_{ia+1} & a < b - a 
\end{cases}
\]

This means that the majority rules: if there are more returning rather than starting lecturers, then as soon as the last r-type lecturer has done her duty, the transport leaves; if there are more s-type lecturers, the transport departs at the requested time of the earliest starting lecturer. As long as the min-max objective function is considered, the optimal time is given by

\[
t_{OPT} = \begin{cases} 
  r_{i1} + s_{i1} & r_{ia} \leq r_{i1} + s_{i1} \leq s_{ia+1} \\
  r_{ia} & r_{i1} + s_{i1} < r_{ia} \\
  s_{ia+1} & r_{i1} + s_{i1} > s_{ia+1} 
\end{cases}
\]

The ideal departure time, minimizing \( \max\{ (t - r_{i1}), (s_{i_b} - t) \} \), is the average of earliest and latest requested times. However, the departure time has to satisfy \( r_{ia} \leq t \leq s_{ia+1} \) since the transport cannot leave before the last returner has finished teaching (\( r_{ia} \)), or after the earliest time of a starting lecturer (otherwise she will be late). If the average is larger than \( r_{ia} \) or smaller than \( s_{ia+1} \), the optimal time for departure will be \( r_{ia} \) or \( s_{ia+1} \) respectively.

Note that for both objective functions the optimal departure time of a given one-directional transport is completely determined. We are then left only with searching for the optimal partition.

### 2.2 Partitioning

The notations and definitions of finding an optimal partition follow Chang et al. [5].

Given a linear ordered set \( \Theta = \{ \theta_1 < \theta_2 < \ldots < \theta_N \} \), a partition is a collection of pairwise disjoint subsets \( \Pi = (\pi_1, \pi_2 \ldots, \pi_k) \) whose union is \( \{1, 2, \ldots, N\} \). The partition size is \( |\Pi| = k \). Let \( n_i = |\pi_i| \), the sequence \( (n_1, n_2, \ldots, n_k) \) is referred to as the partition shape. A consecutive partition is one whose all partition parts, \( \pi_i \), consists of consecutive integers. Let \( f \) be a function defined on all possible partitions. The
optimal partition problem is that of finding a partition which minimizes or maximizes $f$.

When considering the TCTP, we search for a partition of the set of times $S \cup R$. The partition size $k$ represents the number of transports. Each partition part represents one transport, with partition size being equal to the number of passengers in the relevant transport. If the $j$th transport is implemented using a vehicle of capacity $C_j$, the $j$th partition size must evidently satisfy $n_j \leq C_j$.

2.3 Definition of a solution

A solution for the one-directional problem is a pair $(\Pi, T)$ consists of a partition of $S \cup R$, $\Pi = (\pi_1, \pi_2, \ldots, \pi_k)$, and a sequence of departure times $T = (t_1, t_2, \ldots, t_k)$, with $t_i$ denoting the departure time of transport $\pi_i$ for all $i = 1, 2, \ldots, k$. We assume, without loss of generality, that for a given solution $t_1 \leq t_2 \leq \ldots \leq t_k$.

In section 2.1 we have shown how to define the departure times once the partition is given. Hence, for the one-directional case, $\Pi$ determines $T$. This will no longer be true for the two-directional case, because at time $t_{OPT}$ the car may not be available. In section 4 we shall present an algorithm for finding all the linear orders of $S \cup R$, so that the optimal partition is a consecutive one. If only consecutive partitions of an ordered set $\Theta$ are considered, the partition shape completely describes the partition. Given a linear ordered set $\Theta$ and a partition shape $(n_1, \ldots, n_k)$, the transport arrangement is set, with $n_1$ passengers on the first transport, $n_2$ on the second, etc. We still need to find the optimal order-and-shape for a given objective function.

3 The simple one-directional case

The simple case is one with either $R = \emptyset$ or $S = \emptyset$ or $s_n < r_{n+1}$. In the latter scenario the lecturers’ wish-list divides into two separate lists, the s-type and the r-type lists (a mixed transport cannot be applied). For all three scenarios, given a time-list of size $N$, one should first arrange the list in increasing order. If lecturer $i$ have asked to depart earlier than lecturer $j$, it does not make sense to arrange the $j$th lecturer before the $i$th one (all lecturers have equal priorities). The optimal solution is than given by a consecutive partition. For $k$ transports and vehicles of a same capacity $C$, the optimal partition problem can be solved by a reduction. Chakravarty et al. [4] have shown that if $f$ is a minimum additive objective function, whose optimal partition is known to be consecutive, the optimal partition is than found by a reduction to the shortest path problem [1,8]. A directed graph with vertices $\{1, 2, \ldots, N, N + 1\}$ and arcs $(i, j)$, with $i < j$, is built. A weight $f(\{i, i + 1, \ldots, j - 1\})$ is assigned for each arc $(i, j)$. A directed path from 1 to $(N + 1)$ represents a possible consecutive partition. The capacity limit $C$ is expressed by including the arc only if $j - i \leq C$. Since the maximal partition size is restricted by $k$, the number of transports, the path should not consist of more than $k$ arcs. One should than use the Bellman-Ford algorithm [1] and stop the procedure after $k$ iterations. A simple exercise in Bellman-Ford’s algorithm shows that the reduction can also be applied for the min-max objective function.

Consider an example with $N = 7$ s-type lecturers. In this example the time-list is given by $S = \{7, 9, 10, 11, 13, 15, 15\}$ ($R = \emptyset$), the number of transports is $k = 3$, and each transport has a maximal capacity of $C = 3$ passengers. The reduction to the
shortest path problem is illustrated in Figure 1. Each vertex represents a single lecturer in \( S \), except for vertex 8, which represents \( N + 1 \). The other vertices are numbered according to their order in \( S \). Additionally, each vertex is placed along a time-line to demonstrate the values in \( S \).

![Shortest path problem diagram](image)

**Fig. 1** The reduction to the shortest path problem for a simple one-directional example.

The weights of the arcs are given according to the chosen objective function. Figure 2 depicts the resulting weighted graphs when considering the objective functions \( f_{\text{Additive}} \) and \( f_{\text{MinMax}} \). Note that the resulting shortest path (i.e., optimal partition) is different for each objective function.

![Weighted graph and resulting shortest path](image)

**Fig. 2** The weighted graph and resulting shortest path for a simple one-directional example with the objective functions: (a) \( f_{\text{Additive}} \) (b) \( f_{\text{MinMax}} \).
The depicted reduction is applicable whenever the capacity \( C \) is the same for all vehicles. A problem arises when vehicles of different capacities, \( \{C_j\} \), are considered, since the number of outgoing edges from each vertex in the reduction’s graph is no longer unambiguous. One may attempt solving this problem by adding edges with respect to the maximal possible capacity, i.e., drawing the graph for a uniform capacity scenario with \( C = \max\{\{C_j\}\} \). The other smaller capacities are reflected in the reduction by imposing additional constraints, during the search for the shortest path, which are related to the vehicles that have already been used. However, this solution also fails, since Bellman-Ford, as well as other shortest path algorithms, relies on the optimal-substructures property \cite{7}. According to the property, subpaths of shortest paths are also shortest paths. This clearly does not hold when the aforementioned constraints are imposed. Nevertheless, when the capacities’ variance is small, solving a uniform problem with \( C = \max\{\{C_j\}\} \) might practically work. For a large enough \( k \), many of the possible arrangements will still be valid, hopefully including the optimal ones.

Another inapplicable scenario is one in which the passengers are prioritized. Adding priorities inherently contradicts the consecutiveness of the optimal partition, which is an essential precondition of the simple one-directional case. Nevertheless, one may think of a weakened notion of prioritization, in which the priorities do not change the order of the passengers. An organization may decide, for the sake of appearance, not to allow a senior lecturer to take the place in a transport of a junior lecturer that finished work before her, but at the same time find an arrangement in which senior lecturers generally wait less time than others. In this case the priorities only affect the weights of the edges, so the reduction clearly holds.

4 The complex one-directional case

The case with \( R, S \neq \emptyset \) and \( s_0 \geq r_0+1 \) (i.e., the last s-type lecturer does not depart earlier than the first r-type lecturer) is rather complex. Unlike the previously discussed simple case, the optimal partition, for the case of uniform capacity \( C \), is not necessarily obtained by a consecutive partition of the increasingly ordered set \( S \cup R \). We demonstrate this with an example.

Consider \( S = \{8, 13, 14, 19\} \) and \( R = \{16, 21, 21\} \), with \( k = 3 \) transports and vehicle capacity of \( C = 3 \) passengers. Arranging \( S \cup R \) with increasing order stems \( \Theta_0 = \{\underbar{S}, \underbar{13}, \underbar{14}, \underbar{19}\} \). Rightward and leftward arrows denote s-type and r-type lecturers respectively. The only possible consecutive partition is \( (\pi_1, \pi_2, \pi_3) = ((1, 2, 3), (4, 5), (6, 7)) \). This partition describes the transports \( (\underbar{S}, \underbar{13}, \underbar{14}), (\underbar{16}, \underbar{19}), \) and \( (\underbar{21}, \underbar{21}) \), with objective functions \( f_{\text{Additive}} = 14 \) and \( f_{\text{MinMax}} = 6 \). However, better partitions exist. The additive optimal partition is \( \{\{1\}, \{2, 3, 5\}, \{4, 6, 7\}\} \), describing the transports \( (\underbar{S}), (\underbar{13}, \underbar{14}), (\underbar{16}, \underbar{19}), (\underbar{21}, \underbar{21}) \), with \( f_{\text{Additive}} = 12 \). The min-max optimal partition has \( f_{\text{MinMax}} = 5 \), with the partition given by \( \{\{1, 2\}, \{3, 5\}, \{4, 6, 7\}\} \), describing the transports \( (\underbar{S}, \underbar{13}), (\underbar{16}, \underbar{19}), (\underbar{21}, \underbar{21}) \).

Both optimums are non-consecutive with respect to the set \( \Theta_0 \), but do form a consecutive partition with respect to \( \Theta_1 = (\underbar{S}, \underbar{13}, \underbar{14}, \underbar{19}, \underbar{21}, \underbar{21}) \). Figure 3 displays the reduction and resulting shortest path for the additive objective function, \( f_{\text{Additive}} \). The vertices in the figure are numbered according to \( \Theta_1 \). Note that some vertices have outdegree or indegree smaller than \( C \), due to the s-r restriction.

In the spirit of the above example we suggest the following procedure for the complex case. The set \( S \cup R \) should first be arranged in a manner in which the optimal
solution is achieved by a consecutive partition. As discussed in section 3, when the optimal partition is indeed consecutive, it can be found using a polynomial-time reduction.

4.1 An algorithm for solving the complex case

We define an order relation, $\prec$, over the set $S \cup R$ by

- for $s_i, s_j \in S, s_i < s_j \rightarrow s_i \prec s_j$
- for $r_i, r_j \in R, r_i < r_j \rightarrow r_i \prec r_j$
- for $s_i \in S$ and $r_j \in R, s_i < r_j \rightarrow s_i \prec r_j$

According to the first condition, if an $s$-type lecturer should depart before another $s$-type lecturer, she will appear prior on the list. The second condition implies the same for $r$-type lecturers. The last condition says that if an $r$-type lecturer wants to depart after an $s$-type lecturer, then the $r$-type will also appear after the $s$-type on the list (note that the two cannot share a transport). As oppose, even if an $r$-type lecturer finishes teaching before the requested departure time of an $s$-type lecturer, it is still possible for her to depart after the starting lecturer. It is always possible, in principle, to send all the returning lecturers at the last transport (though it is usually not an optimal arrangement).

**Definition 1** An order of $S \cup R$ which is a linear extension of the order $\prec$ is called a proper order.

According to the following theorem, the proper orders and their consecutive partitions completely describe the problem, in the sense that the best consecutive partition of one of the proper orders, is the optimal partition.

**Theorem 1** Given a set of times $S \cup R$, there exist a proper linear order, $\tilde{\prec}$, and a partition $\Pi$ which is optimal (with regards to any of the two mentioned objective functions), so that $\Pi$ is a consecutive partition with respect to $\tilde{\prec}$.

The proof of theorem 1 is given in section 4.2.

We now present an algorithm for solving a uniform capacity one-directional complex case:
1. Given a time list $S \cup R$, create all its proper linear extensions. Algorithms for generating the linear extensions of a partial order exist in the literature [12,13].

2. For each proper order, build the reduction graph by the method described in section 3, keeping in mind that some arcs are impossible due to the s-r restrictions (in a specific transport all the s-type requested times should not be later than the r-type times).

3. Find the shortest path in the graph, by a Bellman-Ford algorithm, restricted to $k$ steps. That shortest path describes the best $k$ transports arrangement for the given order.

The algorithm efficiency depends on several factors: (i) the number of linear extensions of the given order; (ii) the efficiency of linear extensions generation; and (iii) the efficiency of finding the best solution described by the shortest path.

As for the last issue, a Bellman-Ford algorithm finds the optimal path in polynomial time $O(Nk^2)$. The generation of linear extensions is also efficiently done by the known algorithms, in time which is linear in the number of extensions. The efficiency is disturbed only by the number of linear extensions. As close as the given case is to the simple one, the number of extensions will be relatively small. The whole algorithm is then efficient. On the opposing side lies the worst case scenario – when the last returner should depart before the first starting lecturer. When this occurs, every linear order, which is S-ordered and R-ordered, is a proper linear extension. For an even $N$, the number of extensions is maximal if $n = m = N/2$ (the number of r-type equals the number of s-type lecturers). Then, the number of extensions is $\binom{n}{n/2}$, which is exponential in $n$. The algorithm will then be restricted to $n$ of limited size. In addition, the algorithm is wasteful in the sense that some solutions might be generated more than once. The extreme case is when $k = N$. The optimal solution for this case is to send a lecturer per transport. This solution will be repeated in all of the generated linear extensions.

4.2 Definitions extension and proof of theorem 1

We summarize and extend the previously given notations and definitions. The subset collection $P = (\pi_1, \pi_2, \ldots, \pi_k)$, denotes a partition of size $k$, with $\bigcup_{i=1}^{k} \pi_i = \{1, 2, \ldots, N\}$. To keep the notations simpler, we shall identify $P$ with a partition of the actual requested times. We then define $S_i = S \cap \pi_i = \{s_{i1}, s_{i2}, \ldots, s_{i\sigma_i}\}$, where $\sigma_i = |S_i|$ is the number of s-type passengers in the $i$-th transport, and, by definition, $s_{i1} \leq s_{i2} \leq \ldots \leq s_{i\sigma_i}$. Similarly, for the r-type passengers, we have $R_i = R \cap \pi_i = \{r_{i1}, r_{i2}, \ldots, r_{i\rho_i}\}$, $\rho_i = |R_i|$, and $r_{i1} \leq r_{i2} \leq \ldots \leq r_{i\rho_i}$.

**Definition 2** (i) If $r_{i\rho_i} \leq s_{i1}$ for every partition part $\pi_i$ (where $i = 1, 2, \ldots, k$) the partition is said to be **feasible**.

The feasibility of a partition guarantees that the last returning lecturer does not finish after the requested time of the first starting lecturer.

(ii) A solution $(P, T)$ is called a **feasible solution** if $r_{i\rho_i} \leq t_i \leq s_{i1}$. If $(P, T)$ is a feasible solution, the partition is also feasible. In addition, the time of departure is possible.
We define the objective functions of section 2.1 explicitly. Given a solution \((\Pi, T)\), the additive objective function is

\[
 f_{\text{Additive}}(\Pi, T) = \sum_{i=1}^{k} f_A^{(i)}(\pi_i, t_i)
\]

(3)

where

\[
 f_A^{(i)}(\pi_i, t_i) = \sum_{r \in \pi_i} (t_i - r) + \sum_{s \in \pi_i} (s - t_i)
\]

(4)

The min-max objective function is similarly defined by

\[
 f_{\text{MinMax}}(\Pi, T) = \max_{1 \leq i \leq k} \{ f_M^{(i)}(\pi_i, t_i) \}
\]

(5)

where \( f_M^{(i)}(\pi_i, t_i) \) is the largest waisted time in the \(i\)-th transport, and is given by

\[
 f_M(\pi_i, t_i) = \max\{\max_{s \in S_i}\{s - t_i\}, \max_{r \in R_i}\{t_i - r\}\}
\]

(6)

We can always take the set \( T \) to be the optimal set, as given in equations (1) and (2). Nevertheless, for the sake of generality and proof simplicity, we present the definitions for a general possible sequence of departure times \( T \).

**Definition 3** A feasible solution \((\Pi, T)\) defines the following ordered set

\[
 \{r_{11}, r_{12}, \ldots, r_{1\rho_1}, s_{11}, s_{12}, \ldots, s_{1\sigma_1}, r_{21}, r_{22}, \ldots, r_{2\rho_2}, s_{21}, s_{22}, \ldots, s_{2\sigma_2}, \ldots, r_{k1}, r_{k2}, \ldots, r_{k\rho_k}, s_{k1}, s_{k2}, \ldots, s_{k\sigma_k}\}
\]

This order is called the **canonical order** of \( S \cup R \) defined by \((\Pi, T)\).

**Definition 4** Given a partition \( \Pi \) and two requested times, \( s \) and \( s' \), we define an \((s, s')\)-**swap**, as the action of exchanging \( s \) and \( s' \) in the corresponding parts of \( \Pi \). The partition obtained by an \((s, s')\)-swap, is denoted by \( \Pi^{s \leftrightarrow s'} \). Explicitly, if \( s \in S_i \) and \( s' \in S_j \), we define the parts of \( \Pi^{s \leftrightarrow s'} = \{\pi_1^{s \leftrightarrow s'}, \ldots, \pi_k^{s \leftrightarrow s'}\} \) by

\[
 \pi_i^{s \leftrightarrow s'} = \begin{cases} 
 (\pi_i \setminus \{s\}) \cup \{s'\} & \text{if } l = i \\
 (\pi_j \setminus \{s'\}) \cup \{s\} & \text{if } l = j \\
 \pi_j & \text{else} 
\end{cases}
\]

(7)

Let \( \Pi \) be a feasible partition. It does not follow that \( \Pi^{s \leftrightarrow s'} \) is also feasible. A swap is called a **feasible swap**, if \( \Pi^{s \leftrightarrow s'} \) is a feasible partition. We similarly define \( \Pi^{r \leftrightarrow r'} \) for two given returning times \( r \) and \( r' \).

**Lemma 1** Let \((\Pi, T)\) be a feasible solution. Let \( s \) and \( s' \) be two starting times such that \( s \in S_i, s' \in S_j, i < j \) and \( s \geq s' \). Then

1. \((\Pi^{s \leftrightarrow s'}, T)\) is also a feasible solution.
2. \( f_{\text{MinMax}}(\Pi^{s \leftrightarrow s'}, T) \leq f_{\text{MinMax}}(\Pi, T) \).
3. \( f_{\text{Additive}}(\Pi^{s \leftrightarrow s'}, T) = f_{\text{Additive}}(\Pi, T) \).
Proof 1. The inequalities for the returning times are immediate because \((\Pi,T)\) is feasible. Hence, we should only show that \(t_i \leq s'\) and \(t_j \leq s\). The first inequality is valid since \(i < j \Rightarrow t_i \leq t_j\) and \((\Pi,T)\) is feasible, therefore \(t_j \leq s'\). This latter inequality together with \(s \leq s'\) leads to \(t_j \leq s\).

2. Because of the swap, the waisted times of the \(s\) and \(s'\) lecturers change from \(s - t_i\) and \(s' - t_j\) to \(s - t_j\) and \(s' - t_i\), respectively. Anyway, the maximal time out of the four possibilities is \(s - t_i\). Therefore the maximal waisted time has not been increased as a result of the swap.

3. \(f_{\text{Additive}}(\Pi^{\text{extra}},T) = f_{\text{Additive}}(\Pi,T) - (s - t_i) + (s - t_j) - (s' - t_j) + (s' - t_i) = f_{\text{Additive}}(\Pi,T)\).

Definition 5 A partition \(\Pi\) is said to be S-ordered if for any pair \(\pi_i, \pi_j \in \Pi\), \(i < j \Rightarrow s \leq s'\), where \(s \in S_i\) and \(s' \in S_j\). We similarly define an R-ordered partition.

Corollary 1 An optimal partition which is S-ordered and R-ordered, always exists.

Proof Let \((\Pi,T)\) be an optimal partition. If the partition is not S-ordered then there exist \(\pi_i, \pi_j \in \Pi\), with \(i < j\), and two elements \(s \in S_i\) and \(s' \in S_j\), such that \(s \geq s'\). According to Lemma 1 the solution \((\Pi^{\text{extra}},T)\) is feasible and optimal for the two discussed objective functions. If that new partition is still not S-ordered, we continue swapping until an S-ordered partition is obtained. Similarly we lead the partition to its R-ordered form.

We are now ready to give the proof for theorem 1.

Proof (Theorem 1) Let us take the canonical order defined by an optimal solution \((\Pi,T)\), with an S-ordered and R-ordered partition (according to corollary 1 such solution exists). The partition is obviously consecutive with shape \((\rho_1 + \sigma_1, \rho_2 + \sigma_2, \ldots, \rho_k + \sigma_k)\). We prove that this canonical order is a proper order, i.e., if we take \(\pi_i, \pi_j \in \Pi\), then any two elements \(q \in S_i\) or \(R_i\), \(q' \in S_j\) or \(R_j\), are ordered with respect to \(<\). The partition is S-ordered and R-ordered, therefore we only have to check the third condition of \(<\). We take two elements \(s \in S_i\) and \(r \in R_j\) such that \(s < r\). We should prove that \(i < j\). Since \((\Pi,T)\) is a feasible solution \(i \neq j\). We assume indirectly that \(i > j\). Then, \(s \geq t_i \geq t_j \geq r\) in contradiction to \(s < r\).

5 Discussion

We have mathematically modeled the TCTP, and gave an algorithm for solving the one-directional uniform case. The presented problem is a special variation of a DARP. In a general DARP the number of destinations is usually larger than two. Then, besides the need to arrange the partitions and the departure times, a major concern is the arrangement of routes which minimize the cost (or ride times). Our attitude is to find the best arrangement for a fixed number of transports, \(k\), and the cost is evidently in proportion to \(k\). We define a time-window, by treating "s-type" and "r-type" requests differently, but rather than taking the time window parameter (either deadline or release times) as a constraint, we inject it to the objective function, to reflect the dissatisfaction of the customers. The common objective function that DARP researchers usually take refers to the total customer ride times. The fact that the TCTP has only one (forward and backward) route, enables us to treat the dissatisfaction more gently.
The models and methods in use for DARP problems reflect the complexity of the problem. It is common to use local search algorithms. For the TCTP, if the size of solution space increases considerably, local search might be relevant as well. Our presented work provides a natural definition of a neighborhood for a search algorithm. Two steps should be considered – the first is the swap as defined in equation 7. Given a solution $\Pi, T$, one may consider the swapped partitions $(\Pi^s \leftrightarrow s', T)$, $(\Pi^r \leftrightarrow r', T)$, and also $(\Pi^r \leftrightarrow s, T)$. Another evident step is to change the partition shape, i.e., move a passenger from one transport to an adjacent one. When the two directional case is taken into account, a third step, that of changing the time of departure, might be considered as well, because that time is no longer fixed.

Our presented approach still needs to be extended. First, the main problem remained unsolved: the two-directional case, where car availability must be taken into account. Departure times are no longer explicit, because the car may be unavailable at the time which is optimal with regards to the objective function. Even the one-directional case itself can be made more complex, as we briefly discussed in section 3 by considering classes (priorities) or heterogenous capacities. How hard these problems really are? Obviously the simple one-directional case with uniform capacities is easy, but the question remains open for the other scenarios.

References