Angewandte Mathematik und Optimierung Schriftenreihe
Applied Mathematics and Optimization Series
AMOS # 19(2015)

Anke Stieber, Zhi Eric Yuan, and Armin Fügenschuh

School Taxi Routing for Children with Special Needs

Herausgegeben von der
Professur für Angewandte Mathematik
Professor Dr. rer. nat. Armin Fügenschuh

Helmut-Schmidt-Universität / Universität der Bundeswehr Hamburg
Fachbereich Maschinenbau
Holstenhofweg 85
D-22043 Hamburg

Telefon: +49 (0)40 6541 3540
Fax: +49 (0)40 6541 3672

e-mail: appliedmath@hsu-hh.de
URL: http://www.hsu-hh.de/am

Angewandte Mathematik und Optimierung Schriftenreihe (AMOS), ISSN-Print 2199-1928
Angewandte Mathematik und Optimierung Schriftenreihe (AMOS), ISSN-Internet 2199-1936

School Taxi Routing for Children with Special Needs

Anke Stieber · Zhi Eric Yuan · Armin Fügenschuh

Abstract School transport is essential in rural areas, where up to 80% of users of public transport are pupils on their way to school and back. In this article we are going to focus on a particular field of school transport for children with special needs that even make school taxis necessary. We are going to concentrate on children being taken to school in the mornings. This kind of school taxi routing is completely different from ordinary school bus routing in rural areas. First of all, vehicles used for this kind of routing would have to provide specially designed spaces to possibly place and transport wheelchairs of different types, as well as their owners. School taxis would collect pupils with special needs from their homes in the mornings in order to take them to school. For each school a time window would be given outside of which the school taxis would not be allowed to drop the pupils at the school. With this, we are faced with the classical vehicle routing problem with time windows generalized by picking up, in our case, children with special needs, and delivering them.

We are addressing real-world instances by applying a new heuristic algorithm which is an iterative construction approach combined with a parametrized greedy metaheuristic (PGreedy). PGreedy is a parametrization of greedy scoring functions where good weights are obtained by parameter optimization.

A. Stieber*
Tel.: +49-40-6541-2755
E-mail: anke.stieber@hsu-hh.de

Z. E. Yuan*
Tel.: +49-40-6541-3241
E-mail: yuanz@hsu-hh.de

A. Fügenschuh*
Tel.: +49-40-6541-3540
E-mail: fuegenschuh@hsu-hh.de

*Helmut Schmidt University / University of the Federal Armed Forces Hamburg
Holstenhofweg 85, 22043 Hamburg, Germany

Keywords routing · scheduling · metaheuristic · integer linear programming

1 Introduction

In Germany school attendance is compulsory for all children. The federal states of Germany are responsible for all regulations connected with school and education as well as school transport. In general school transport is integrated in public transport and therefore organized by counties and district towns. Most school transport is done with school buses, especially in rural areas. Here school transport is part of public transport with up to 80% of all passengers being pupils, see Fiedler [14]. A minority of pupils is transported by so called special purpose school buses. In order to reduce costs both with respect to organizing the transport and the transport as such, and in order to save tax money, the development of a computerized optimization system for school bus routing and scheduling is currently more and more in demand. Here, we focus on a special field of school transport, namely school transport for pupils with special needs with the help of school taxis.

The routing of school taxis for children and young persons with special needs is completely different from school bus routing in general. Pupils who take a school bus in the morning gather to their neighboring bus stop and wait for a bus to take them to school. In the afternoons a school bus takes them from school back to their neighboring bus station. Usually, school buses are deployed for the first and second classroom lessons in the mornings and for the last two lessons in the afternoons. By contrast to this, school taxis pick up pupils with special needs directly at their homes and transport them to their school. In the afternoons the pupils are carried back to their homes. Often, children with special needs have to travel a very long distance to get to one of the few schools that provide for their special needs.

In our application, vehicles with special designed spaces for different types of wheelchairs are required for some pupils. Several types of taxis controlled by different taxi companies are available. The vehicles are characterized by a certain passenger capacity and a certain wheelchair capacity. Some of them are even capable of transporting different types of wheelchairs. In total, there are about one hundred different types of vehicles. However, most of the vehicles differ only slightly and therefore can be grouped into classes. The main differences between the vehicles from different classes are their capacity and costs. The passenger capacity of vehicles ranges from 2 to 28 persons. Whereas on some cars wheelchairs are not admissible at all, some vehicles can take from 1 up to 4 wheelchairs. Note that we consider two types of wheelchairs: a normal wheelchair and an electronic wheelchair. The difference between both types is, that the normal wheelchair can be folded up so that it takes less space during transport, while the electronic wheelchair cannot. Thus, an electronic wheelchair takes about twice the space of a normal one. In practice we may assume that the space taken up by one electronic wheelchair equals the space of two normal wheelchairs, and vice versa. Vehicles that have different capac-

ities result in different transportation costs. In general, the cost of a taxi is composed of two parts, the basic cost of a car being provided and the driving cost per kilometer. In addition, some pupils need to be accompanied by an accompanying person on their way to school. Usually, the cost for an accompanying person is fixed independent of the vehicle and the distance covered. Further attributes such as the different speed potential of the vehicles are not considered here.

School taxi transport is organized as follows. In the mornings several school taxis spread out to pick up the pupils directly at their homes and transport them to school, so that they arrive a few minutes before school starts. In the afternoons, when classes have finished, a fleet of school taxis is sent to take the pupils back home. Considering the morning situation, when a pupil is picked up, the vehicle does not need to go to the respective school immediately. If capacity for passengers and wheelchairs - if needed - is available, the taxi may pick up some more pupils before heading for school. Note, that we do not restrict pupils from the same school to be on the same vehicle at the same time. Allowing pupils from different schools to be on board of the same vehicle may makes routing more flexible, which may result in further potential savings of costs.

Several time restrictions are an issue in our application. Driving times and distances between each pupil's home and school are given, and is assumed to be invariant with respect to different school taxis. When a vehicle stops at a pupil's home, a pickup time is specified for each pupil, that is the time the pupil needs to get into the car. A pupil without wheelchair usually takes one minute to get in, while a pupil with a wheelchair needs a longer time, usually three minutes. The drop time at school also depends on the number of wheelchairs on board. Nevertheless we assume a uniform drop time of five minutes for all school nodes. Another time restriction given is that of each pupil being supposed to arrive at school within a certain maximum driving time, which is the maximum time to take the pupil from home to school. Usually, pupils are supposed to arrive at school within a time window of 10 to 15 minutes before school starts. Combined with the maximum driving time, a lower and upper bound on the departure time of the pupils can be derived.

For the routing and the scheduling problem we are only considering the morning problem, i.e. the transportation from pupils' homes to their respective schools. It is a more difficult issue than the afternoon issue. The afternoon problem is picking up all pupils at their schools and delivering them back home one by one. This cannot exactly be regarded as being symmetric with the problem of collecting them from their respective homes in the morning. If we go into more detail, it can be observed, that the time restrictions in the afternoons are not as tight as in the mornings, thus, the transportation back home can usually be done with fewer vehicles. The reason for this relaxed time restriction is the school finishing time. While nearly all schools start within a narrow time window at around 8 a.m. in the morning, the schools release their pupils at different times over a wider range in the afternoon. This wider range of finishing times may result in a smaller number of vehicles that are

needed for the transportation of pupils from school to the pupils' homes. A second reason why the morning problem is more difficult than the afternoon one is the traffic congestion, which is usually higher in the mornings than in the afternoons.

The objective of the school taxi routing problem is to minimize its total operational costs. This chiefly means reducing the number of deployed vehicles due to the high basic cost of each vehicle and optimizing the total distance of all vehicles. Note, that the driving distance from the school taxi depot to the first pupil is not considered, neither is the distance from the last school to the depot.

The described problem is an instance of the Heterogeneous Vehicle Routing Problem with Time Windows (HVRPTW). It consists of a fleet of vehicles of different types (capacity) and customer demands have to be satisfied within given time windows. Usually an heterogeneous Vehicle Routing Problem (VRP) is modeled as a multi-commodity flow problem, whereas each vehicle located in its depot represents a commodity in the network. In our case we are additionally faced with pickup nodes, which consists of a certain supply. This supply is described by pupils and accompanying people, that have to be transferred to delivery nodes (schools). This problem is a generalization of the HVRPTW and is called Pickup and Delivery Problem with Time Windows (PDPTW). The given time windows for each request specify, that the pupils are not allowed to arrive outside the given interval at their corresponding schools. If they arrive earlier waiting is permitted until the start of the time window.

One way to give a MILP formulation of the PDPTW is to put a multi-commodity flow formulation in a time expanded network, see Ford and Fulkerson [7]. Another way is to use a multi-commodity flow model with time variables and apply Miller-Tucker-Zemlin [17] constraints to embed the time restrictions. We use a time discretization with a time unit of one minute since our input time data is aligned to one minute. Due to this fine discretization a model formulation in a time expanded network will result in large-scale MILP problems even for small instances. This leads us to the second variant using the Miller-Tucker-Zemlin constraints to include time restrictions. However, exact methods will not be able to solve large-scale problem instances. Then, for those instances we are more interested in a near optimal solution that can be obtained in reasonable time. To this end we present a metaheuristic approach and apply it to our real-world instances that are related to two German counties Aurich and Bentheim. The metaheuristic is a parametrized greedy approach which is based on a iterative construction (route-building) framework. This metaheuristic is already successfully applied to similar vehicle routing problems such as the integrated planning problem of coordinating bus and school starting times and bus schedules by Fügenschuh [8], the vehicle routing problem with coupled time windows by Fügenschuh [9] and the strategic locomotive scheduling problem by Fügenschuh et al. [10].

The remainder of this article is organized as follows. In section 2 we give an overview of the development in the field of vehicle routing problems with time

windows and with respect to pickup and delivery problems of the last decades. Since this field was well studied we only point to an excerpt of the whole set of available publications. We present the mentioned formulation to the heterogeneous PDPTW in section 3. In section 4 we introduce our metaheuristic by explaining the construction phase and the parametrization of different scoring functions. We apply our approaches to two problem instances in section 5, analyze the results and finally provide some conclusions in section 6.

2 Survey of the Literatur

One of the best-known routing problem in the literature is the Traveling Salesman Problem (TSP). A salesman has to visit a number of cities and thereafter must return to his starting location. The aim is to construct a route, that minimizes the traveled distances. For a survey on TSP we refer to Lawler et al. [15] or Reinelt [20].

The VRPTW is a Multiple Traveling Salesman Problem (MTSP) where a demand and a time window is associated with each city. Vehicles have a certain capacity and demands are to be delivered within given time windows (Crainic and Laporte [3], Toth and Vigo [23]). Whether there are all vehicles from the same type or from different a classification of the VRPTW into homogeneous and heterogeneous can be made.

The VRPTW problem class was well studied in the last decades. Koskosidis et al. [24] addressed the problem concerning soft time windows. Bräysy and Gendreau [2] tackled the VRPTW from the metaheuristic side. They presented a comprehensive overview of different tabu search algorithms as well analyzed and compared the different techniques for Solomon's [22] benchmark test problems. Hashimoto et al. [11] had a look at the VRP with flexible time windows and flexible traveling times. Here, the time constraints were also handled in a soft manner and therefore treated as cost functions. Route generation was done by local search and the determination of the optimal starting times afterwards, which is NP-hard, was realized by a dynamic programming approach.

There is another vehicle routing problem considering pickup and delivery, called VRPPD. VRPPD considers demand, that is transferred between pickup and delivery nodes, but without restricting time windows. A generalization of VRPTW and VRPPD can be regarded as the Vehicle Routing Pickup and Delivery Problem with Time Windows (PDPTW). The application described in section 1 is an instance of such a problem.

Savelsbergh and Sol [21] (1995) made the first attempt to cover all possible generalizations of the VRPPD in a unified notion and presented an overview of existing solution methods. Dumas et al. [5] dealt with the PDPTW and presented a column generation approach that uses constrained shortest path as subproblem. Hosny and Mumford [12] addressed the multiple vehicle PDPTW and presented four construction heuristics, which differ in the manner of iterative or parallel construction and in the way the next solution component is

selected. The iterative version seemed to be the most favorable one according to their computational results. Lu and Dessouky [16] worked on a construction heuristic, which not only takes into account a distance increase to the evaluation criterion but also time window slack reduction and visual attractiveness. The VRPPD was also addressed by Parragh et al. [19]. They proposed a comprehensive survey on routing problems with pickups and deliveries. Formulations for different variants of pickup and delivery problems were given and even time windows and maximum travel times were taken into account. For an extensive overview of the state of the art in exact, heuristic and metaheuristic approaches for several versions of pickup and delivery problems we refer to this article. In addition, they discussed the dynamic and stochastic versions of pickup and delivery problems.

Desrochers et al. [4] present a formulation to the VRPTW and the PDPTW. They present an early survey of exact solution techniques for these problems such as Branch-and-Bound with a set partitioning approach and dynamic programming which appears as subroutines using state space relaxation to compute lower bounds. The authors point to construction heuristics, improvement heuristics and incomplete optimization and address different types of vehicle routing problems.

El-Sherbeny [6] gives a formulation for the VRPTW and different generalizations such as the PDPTW. According to the author exact methods are mainly lagrange relaxation, column generation and dynamic programming, but in some cases they result in very long computation times (several days) to solve fairly small instances. The main part of the article deals with recently published heuristics and metaheuristics. El-Sherbeny summarizes route-building heuristics (generate a solution from the scratch) and route-improving heuristics (search the neighborhood of a solution to find a better one) as well as neighborhood operators for the VRPTW. He also describes and compares some powerful metaheuristics such as simulated annealing, tabu search and genetic algorithms.

Some of the most useful application of the VRPTW contain bank deliveries, postal deliveries, school bus routing and national franchise restaurant service. The school bus routing problem as a VRPTW is addressed by Braca et al. [1]. They made a good attempt to describe the problem of school bus routing in New York City including given data and constraints.

Nanry and Barnes [18] address the PDPTW with a homogeneous vehicle fleet. They proposed a reactive tabu search algorithm, that incorporated three distinct move neighborhoods with act on the dominance of the precedence and coupling constraints. A hierarchical search is employed to change between neighborhoods in order to handle different regions of the solution space. To evaluate their approach they constructed a new set of benchmark problems based on Solomon's benchmark sets.

3 Formulation of the School Taxi Routing Problem

We present a formulation of the school taxi routing problem as a mixed-integer linear program (MILP). In general vehicle routing problems with time windows (VRPTW) and pickup and delivery problems with time windows (PDPTW) are modeled as a multi-commodity minimum cost flow problems, where feasibility of the flow is ensured by subtour elimination constraints or by Miller-Tucker-Zemlin constraints, see Desrochers et al. [4]. In the first case the number of subtour elimination constraints is exponentially in the number of nodes and therefore impractical to solve, we present a formulation using the Miller-Tucker-Zemlin conditions, whereas the number of constraints is polynomially bounded.

3.1 Instance Data

Here, we present the instance data to set up the model formulation for the problem of school taxi routing of pupils with special needs.

We denote the set of all available *vehicles* or *school taxis* by $\mathcal{K}$. The *pickup nodes* $\mathcal{V}_\mathrm{P}$ describe the location of all pupils homes. The *delivery* or *school nodes* $\mathcal{V}_\mathrm{S}$ depict the location of the schools, where pupils have to be transported to. Then, we need each school nodes to be decomposed into several nodes at the same geographic location. This is done in a way, that there is a school node for each pupil attending this school or equivalently for each given transport request of this school. That means after decomposition there is a one-to-one correspondence between pickup nodes $i \in \mathcal{V}_\mathrm{P}$ and delivery nodes $j \in \mathcal{V}_\mathrm{S}$, in a way that $j = i + n$, where $n := |\mathcal{V}_\mathrm{P}| = |\mathcal{V}_\mathrm{S}|$. The same structure is used by Desrochers et al. [4] on page 68 for the PDPTW formulation.

In addition, we have a set of *depots* $\mathcal{V}_\mathrm{D}$ and a complete set of all *nodes* $\mathcal{V} := \mathcal{V}_\mathrm{P} \cup \mathcal{V}_\mathrm{S} \cup \mathcal{V}_\mathrm{D}$. Each vehicle starts and ends the transportation in its own depot. We split each depot node in an *origin depot* node and a *destination depot* node, so that each vehicle k has a origin depot $d_k \in \mathcal{V}_\mathrm{D}$ and a destination depot $a_k \in \mathcal{V}_\mathrm{D}$. We denote the set of *arcs* (roads) connecting pupils home locations, schools and depots by $\mathcal{A} := (\mathcal{V}_\mathrm{D} \times \mathcal{V}_\mathrm{P}) \cup \mathcal{A}_\mathrm{S} \cup (\mathcal{V}_\mathrm{S} \times \mathcal{V}_\mathrm{D})$, whereas the set of *service arcs* $\mathcal{A}_\mathrm{S} \subset (\mathcal{V}_\mathrm{P} \cup \mathcal{V}_\mathrm{S}) \times (\mathcal{V}_\mathrm{P} \cup \mathcal{V}_\mathrm{S})$ contains only those arcs, that correspond to feasible trips between pickup and school nodes. Waiting at a pickup or delivery node is permitted for the vehicles. To model waiting we introduce a loop at each pickup and school node, that has a distance of zero units and a travel time of one minute. Note, that each vehicle first visits a pick up node and then its corresponding school node. Therefore we assume, that all arcs leaving a depot node enter a pick up node, i.e. for all $v \in \mathcal{V}_\mathrm{D}$ we have $\delta^+(v) \in \mathcal{V}_\mathrm{P}$. On the other hand we further assume, that all arcs entering a depot node leave a school node, i.e. for all $v \in \mathcal{V}_\mathrm{D}$ we have $\delta^-(v) \in \mathcal{V}_\mathrm{S}$. We denote the set of pupils transportation requests by the following set of pairs $\mathcal{R} \subseteq \mathcal{V}_\mathrm{P} \times \mathcal{V}_\mathrm{S}$.

For each vehicle $k \in \mathcal{K}$ the passenger capacity is c_k and the wheelchair capacity is $\tilde{c}_k$. Each request $(r, s) \in \mathcal{R}$ provides a supply at its corresponding pickup node $r \in \mathcal{V}_{\mathrm{P}}$, that is the number of passengers p_r to be picked up by a school taxi and transported to school s. The demand at each school node $s \in \mathcal{V}_{\mathrm{S}}$ is denoted by q_s and it is equal to the aggregated supply value of all requests, that consist of a transportation to that certain school s:

$$q_s = - \sum_{r:(r,s)\in\mathcal{R}} p_r.$$

Usually, the supply at a pickup node is one pupil, but there are pupils, who have to be accompanied by a person on their way to school. Furthermore, for each request (r, s) the supply of passengers goes along with the supply of wheelchairs $\tilde{p}_r$, that also have to be transported to school s. In our case we only differentiate two types of wheelchairs, a normal one and an electronic one. Note, that each normal wheelchair counts as one unit, while an electronic wheelchair counts as two units of a normal wheelchair. The corresponding wheelchair demand at a school node $s \in \mathcal{V}_{\mathrm{S}}$ is denoted by $\tilde{q}_s$, with

$$\tilde{q}_s = - \sum_{r:(r,s)\in\mathcal{R}} \tilde{p}_r.$$

For a vehicle $k \in \mathcal{K}$ driving on the arc $(i, j) \in \mathcal{A}$ the transportation cost is denoted by $c_{i,j,k}$. In the case $i, j \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}}$, $c_{i,j,k}$ is given by the distance of the arc (i, j) multiplied with the kilometer price of vehicle k. For $i \in \mathcal{V}_{\mathrm{D}}$, $c_{i,j,k}$ is given by the basic cost of vehicle k, whereas $c_{i,j,k} = 0$ for $j \in \mathcal{V}_{\mathrm{D}}$. Note that there are no arcs between depot nodes.

Given an arc (i, j) we denote the transportation time between node i and node j by $\delta_{i,j}$. If j is visited directly after i and $(i, j) \in \mathcal{R}$, then it will consist of the pick up time (one minute for a pupil without wheelchair and three minutes for a pupil together with a wheelchair), the driving time and the drop time. For reasons of simplification the drop time is set to five minutes independently of dropping wheelchairs or not. For a path starting in i and finishing in j the transportation time is accumulated over all arcs belonging to that path with respect to the corresponding pickup and drop times. $\overline{\delta}_{i,j}$ is the maximum transportation time for the request $(i, j) \in \mathcal{R}$, that means the pupil belonging to this request is not allowed to travel to school longer than $\overline{\delta}_{i,j}$.

Considering the time at which each pupil arrives at its requested school $s \in \mathcal{V}_{\mathrm{S}}$, a narrow time window of 10-15 minutes before school start is given. This time window is specified by the earliest possible arrival time $\underline{t}_s$ and the latest possible arrival time $\overline{t}_s$. Since school starting times differ from school to school and may even be different considering pupils of the same school, we introduce a time horizon $\mathcal{T} = [\tau_S, \tau_E]$ for the transportation problem that is about six hours from 4 to 10 am ($\tau_S = 240$, $\tau_E = 600$). Usually school starts around 8 o'clock, thus, even very long transportation due to a long distance between pupils home and school can be taken into account.

Variables

To formulate the school taxi routing problem as an instance of the heterogeneous PDPTW we introduce the following variables.

$$\forall (i,j) \in \mathcal{A},\, k \in \mathcal{K} :\; y_{i,j}^k \in \{0,1\}, \tag{1}$$

where $y_{i,j}^k = 1$ if and only if school taxi k travels from i to j. Furthermore we have some integer variables to model the departure time of the flow at pickup and delivery nodes:

$$\forall i \in \mathcal{V}_\mathrm{P} \cup \mathcal{V}_\mathrm{S} :\; t_i \in \mathbb{N}. \tag{2}$$

The following variables

$$\forall i \in \mathcal{V},\, k \in \mathcal{K} :\; \pi_i^k \in \mathbb{N} \tag{3}$$

and

$$\forall i \in \mathcal{V},\, k \in \mathcal{K} :\; \omega_i^k \in \mathbb{N} \tag{4}$$

are essential for the pickup and delivery part of this vehicle routing problem. We use this variables to represent the passenger load respectively wheelchair load after node i has been serviced by vehicle k. The initial passenger load as well as the initial wheelchair load at depot nodes is set to 0. Again, an electronic wheelchair is replaced by two units of a normal wheelchair.

Objective Function

The objective function is to minimize the total operating costs:

$$\sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{i,j,k} \cdot y_{i,j}^k \quad \to \quad \min. \tag{5}$$

Constraints

Routing of vehicles. Each depot contains only one vehicle and therefore provides at most one unit of flow:

$$\forall i \in \mathcal{V}_\mathrm{D},\, k \in \mathcal{K} :\; \sum_{j\in\mathcal{V}_\mathrm{P}} y_{i,j}^k \leq 1. \tag{6}$$

In order to make sure, that each deployed vehicle returns to its depot at the end of the schedule the following constraints must be satisfied:

$$\forall i \in \mathcal{V}_\mathrm{D},\, k \in \mathcal{K} :\; \sum_{j\in\mathcal{V}_\mathrm{P}} y_{i,j}^k = \sum_{j\in\mathcal{V}_\mathrm{S}} y_{j,i}^k. \tag{7}$$

Each pupil and school node has to be visited exactly once by exactly one vehicle:

$$\forall j \in \mathcal{V}_\mathrm{P} \cup \mathcal{V}_\mathrm{S} :\; \sum_{i\in\mathcal{V}} \sum_{k\in\mathcal{K}} y_{i,j}^k = 1. \tag{8}$$

Flow conservation has to hold for all pickup and delivery nodes (inflow is equal to outflow):

$$\forall j \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}},\, k \in \mathcal{K} :\ \sum_{i \in \mathcal{V}} y_{i,j}^{k} = \sum_{i \in \mathcal{V}} y_{j,i}^{k}. \tag{9}$$

A request (a pair of pickup and delivery node) must be served by the same vehicle, which requires the following condition:

$$\forall j \in \mathcal{V}_{\mathrm{P}},\, k \in \mathcal{K} :\ \sum_{i \in \mathcal{V}} y_{i,j}^{k} = \sum_{i \in \mathcal{V}} y_{i,j+n}^{k}. \tag{10}$$

Scheduling of vehicles. To model the time window restriction for all school nodes, we use the time variables t_i, $i \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}}$:

$$\forall i \in \mathcal{V}_{\mathrm{S}} :\ \underline{t}_i \le t_i \le \overline{t}_i. \tag{11}$$

The feasibility of the schedule is guaranteed by the following time compatibility constraints:

$$\forall (i,j) \in \mathcal{A}_{\mathrm{S}},\, k \in \mathcal{K} :\ t_i + \delta_{i,j} + M \cdot (y_{i,j}^{k} - 1) \le t_j, \tag{12}$$

where M is a sufficiently large constant. These constraints accomplish route connectivity, therefore we refer to the inequalities Miller, Tucker and Zemlin proposed in their TSP formulation in [17] to replace the exponential number of subtour elimination constraints. Miller, Tucker and Zemlin used a special case where $\delta_{i,j} = 1$ and M is the number of nodes in the TSP.

In addition, we have to include time restrictions, that ensure a school node is visited after the corresponding pupil node and the maximum transportation time for each request is not exceeded:

$$\forall i \in \mathcal{V}_{\mathrm{P}} :\ t_i + \delta_{i,i+n} \le t_{i+n}, \tag{13}$$

and

$$\forall i \in \mathcal{V}_{\mathrm{P}} :\ t_i + \overline{\delta}_{i,i+n} \ge t_{i+n}. \tag{14}$$

Loading constraints of vehicles. Two types of loads have to be taken into account, passenger load and wheelchair load. Constraints to satisfy the passenger load compatibility are given by

$$\forall (i,j) \in \mathcal{A},\ \text{where } j \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}},\, k \in \mathcal{K} :\ \pi_i^k + p_j + M_k^{pass} \cdot (y_{i,j}^{k} - 1) \le \pi_j^k, \tag{15}$$

with sufficiently large constants M_k^{pass}. The initial passenger load is set to:

$$\forall i \in \mathcal{V}_{\mathrm{D}},\, k \in \mathcal{K} :\ \pi_i^k = 0. \tag{16}$$

The passenger load at each node is restricted to the vehicle capacity:

$$\forall i \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}},\, k \in \mathcal{K} :\ \pi_i^k \le c_k. \tag{17}$$

Finally the passenger load conditions have to be transferred to the wheelchair load:

$$\forall (i,j) \in \mathcal{A}, \text{ where } j \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}},\, k \in \mathcal{K}:\ \omega_i^k + \tilde{p}_j + M_k^{wheel} \cdot (y_{i,j}^k - 1) \leq \omega_j^k, \quad (18)$$

with sufficiently large constants M_k^{wheel}. The initial wheelchair load is set to:

$$\forall i \in \mathcal{V}_{\mathrm{D}},\, k \in \mathcal{K}:\ \omega_i^k = 0. \quad (19)$$

The wheelchair load at each node is restricted to the vehicle capacity:

$$\forall i \in \mathcal{V}_{\mathrm{P}} \cup \mathcal{V}_{\mathrm{S}},\, k \in \mathcal{K}:\ \omega_i^k \leq \tilde{c}_k. \quad (20)$$

Constraints (12) ensure the route connectivity whereas (15) and (18) guarantee that the capacity of a vehicle is not exceeded throughout the tour. Note, that for all those constraints we need to fix constant values M, M_k^{pass} and M_k^{wheel}. We set the constant values in the following way:

$$M := 600, \quad M_k^{pass} := c_k + 2 \text{ and } M_k^{wheel} := \tilde{c}_k + 1 \quad \forall k \in \mathcal{K}. \quad (21)$$

In general applying Miller-Tucker-Zemlin constraints to vehicle routing problems instead of an exponential number of subtour elimination constraints results in weak linear programming relaxations and thus in poor lower bounds in a Branch-and-Cut algorithm, see Desrochers and Laporte [5]. The reason is, that the Miller-Tucker-Zemlin constraints do not define facets of the underlying polytope of the convex hull of the feasible solution space.

4 Primal Heuristic

The PDPTW is NP-hard and large-scale instances of this combinatorial optimization problem are too difficult to be solved exactly in a reasonable amount of time. In cases where the focus is not only on obtaining a feasible solution but on the quality of the solution, heuristics become important. Heuristic solutions can also be applied to exact methods such as Branch-and-Cut procedures to provide a good initial solution, which speeds up the computation. The result of heuristic methods are not necessarily optimal, but they can be tuned to perform very well.

El-Sherbeny [6] presents an overview of recently developed heuristic methods to the VRPTW. He separates route-building heuristics into iterative and parallel variants. Iterative methods build routes one after another, often they result in poor quality due to assigning the last unrouted customers to a route, which are often scattered over the geographic area. In the contrary parallel route-building heuristics generate several routes simultaneously to overcome this drawback. In general parallel variants perform better compared to iterative variants. As a second class route-improving heuristics are mentioned. Therefore a neighborhood of a solution has to be defined. Checking some or all of the neighboring solutions may expose better solutions regarding the objective function value. This is also called local search. El-Sherbeny presents neighborhood-operators for the VRPTW.

4.1 PGreedy Heuristic

At first, we present a heuristic method constructing a feasible solution based on a greedy approach. This construction (route-building) procedure is an iterative variant and generates routes one by one. When generating a route, a path is built incrementally by appending unserved customers at the end of the current path, while minimizing the driving time (or distance). Relying on one simple greedy scoring function means relying on one criterion, since this is doubtful and unstable, we want to overcome this structural problem. In order to make the greedy approach more robust, we introduce a parametrized greedy algorithm (PGreedy), which was proposed by Fügenschuh [8] as a new type of metaheuristic, while tackling the vehicle routing problem with coupled time windows. At the end of our greedy construction a local search step is conducted which replaces an expensive vehicle type by a cheaper one, if the capacity constraints are satisfied. In the following we give a detailed explanation of our algorithm.

Greedy Construction Framework

We present a greedy construction framework for the school taxi routing problem. The greedy construction heuristic builds a complete solution from scratch. Each route is constructed sequentially by iteratively choosing an immediate best solution component until a complete solution is generated. There are mainly three greedy decisions to be made during the construction, as outlined in Algorithm 1: select a vehicle, select the first node of a route and select every next node.

Depending on our real-world instances there are different types of taxis, which differ in cost and capacity (including passenger capacity and wheelchair capacity). Since a vehicle type exchange local search procedure is applied, the vehicle cost is not our first concern during the construction process. We build a Pareto optimal set of taxis in terms of passenger capacity and wheelchair capacity. In this Pareto set each vehicle type is no worse than any other taxi regarding both capacities. Then, a taxi type is a uniformly selected random element from the Pareto set to initialize a route.

In the next step we select the best pickup node to start the route. A pickup node may serve as the first one if it starts as early as possible, so that the duration of the route can be long. On the other hand, the pupils who have the longest distances to their school should be preferred, so that no detour is needed to pick them up in a later construction phase. A pupil $i \in \mathcal{V}_{\mathrm{P}}$ going to school $s \in \mathcal{V}_{\mathrm{S}}$ has an implicit time window to be picked up, with a lower bound of $\underline{t}_i = \underline{t}_s - \overline{\delta}_{i,s}$, i.e. the start of the schools arrival time window minus the maximum transportation time of pupil i. The upper bound $\overline{t}_i = \overline{t}_s - \delta_{i,s}$, i.e. the end of the schools arrival time window minus the minimum transportation time. The best first pickup node i^* is selected as

$$i^* = \arg\min_{i \in N} \left(\underline{t}_i + \overline{t}_i\right) \cdot \mathrm{rand}\left(\frac{1}{\gamma}, 1\right), \tag{22}$$

where N denotes the set of unselected nodes. At this step N is initialized to $N = \mathcal{V}_\mathrm{P}$ and $\mathrm{rand}(\frac{1}{\gamma}, 1)$ is a noise term that generates a uniformly random number ranging from $\frac{1}{\gamma}$ to 1, where $\gamma \geq 1$ is a parameter that allows candidates of up to γ times worse than the best one to be selected.

After the first pickup node is selected, the *current time* t_C of the current route is defined and set to the earliest possible time the selected first pupil may be picked up. The selected pickup node is removed from the unselected nodes N and the corresponding delivery node is added to N. Then we have to check N for feasibility, because only feasible nodes are allowed to be appended to the route. Let N_{feas} be the set of feasible nodes from N. The following feasibility criteria are checked to find N_{feas}: passenger capacity, wheelchair capacity, and time constraints. In a basic implementation, we allow only pupils of the same school to be on board of a school taxi simultaneously. In such a case, all pupils, belonging to a different school than the school of the pupil recently selected, are not feasible, thus not included in N_{feas}. They become feasible when the current school will be delivered. In order to check for time feasibility of a next pupil node, we need to check whether the taxi is able to reach the current school before the latest school arrival time. In a more advanced implementation when pupils of different schools are allowed to be on board simultaneously, the time feasibility check is more complicated. All permutations of the to be delivered schools have to be checked, to ensure that there is at least one school permutation such that all pupils can be delivered to their schools in time. N_{feas} contains all feasible nodes that can be selected next and to determine the best next node j after node i was appended the following factors are considered:

1. Link time $\Delta t_j := \max\{\delta_{i,j}, \underline{t}_j - t_C\}$, which is the maximum of minimum transportation time between i and j and the difference of the earliest starting time of node j and the current time.
2. Pickup time $t_j^{pickup} \in \{1, 3\}$, depends on the wheelchair supply at j (pupils with wheelchair usually need a longer time to get in the taxi). If the taxi still has space for wheelchairs, pupils with wheelchairs will be preferred.
3. School time $t_j^{school} := \delta_{j,j+n}$ is the transportation time required to get from pupil j's home to his school.
4. All school time t_j^{all} is defined as follows. If pupil j is from a school different from the pupils on board, the transportation time from pupil j's school to the furthest school in the current route is computed, otherwise, it is set to 0. This is to penalize picking up pupils from different schools.

The greedy best next node j^* is selected as follows:

$$j^* = \arg\min_{j \in N_{feas}} \lambda_1 \cdot \Delta t_j + \lambda_2 \cdot t_j^{pickup} + \lambda_3 \cdot t_j^{school} + \lambda_4 \cdot t_j^{all}, \qquad (23)$$

where the four-element vector $\lambda = (\lambda_1, \lambda_2, \lambda_3, \lambda_4)^T$ contains the greedy parameter that are to be determined dynamically by our PGreedy approach described below. Besides updating the current time t_C of the route that is under construction, we also have to update the passenger and wheelchair load

of the route to check for vehicle capacity feasibility. In case a school node is added to the current route, the route respectively school taxi has be unloaded at this location by the number of pupils and wheelchairs that are dropped. A route is finished when there are no more feasible nodes available. In the case N is not empty a new route is started and the construction terminates when all pupils are served, i.e. $N = \emptyset$.

Algorithm 1: Greedy construction for school taxi routing

```
Data: Set of taxis K, set of pickup nodes P, set of delivery nodes S
Result: Set of routes R
Initialize unselected nodes N ← P;
Initialize R ← ∅;
while N is not empty do
    Initialize new route r by selecting a best taxi k ∈ K;
    Select best first pickup node n_0 ∈ N, and append it to route r;
    Remove n_0 from N;
    Add the corresponding delivery node s_0 of n_0 to N;
    while route r is not full and N is not empty do
        Find feasible nodes N_feas ⊆ N;
        Select a best next node n ∈ N_feas;
        Append n to route r;
        if n is pickup node then
            Remove n from N;
            Add the corresponding delivery node s of n to N;
        else
            Unload route r and remove n from N;
    R ← r;
return R;
```

PGreedy Algorithm

Since there are further vehicle routing applications where PGreedy could be applied successfully, see [8], [9] and [10], we use this metaheuristic to solve our PDPTW instances to near optimality in reasonably short time. The crucial point within every greedy algorithm is finding a proper criterion to select local solutions. Such a criterion decides for the search direction.

Our greedy scoring function in equation (23) consists of four criteria, as described in section 4.1. Each criterion has a weight λ_i, $i \in \{1, 2, 3, 4\}$. Thus our parametrized scoring function s_j is a mapping $s_j : \mathbb{Q}^4 \to \mathbb{Q}$, where $j \in N$ is a pickup node or a delivery node. Let us recall the scoring function s_j which is linear in its parameter vector $\lambda \in \mathbb{Q}^4$:

$$s_j(\lambda) := \lambda_1 \cdot \Delta t_j + \lambda_2 \cdot t_j^{pickup} + \lambda_3 \cdot t_j^{school} + \lambda_4 \cdot t_j^{all}. \tag{24}$$

It is obvious that the local selection of elements and hence the entire heuristic solution depends on the choice of $\lambda \in \mathbb{Q}^4$. Thus we have to find an appropriate

vector λ to yield a reasonable good heuristic solution. Let $z(\lambda)$ be the objective function value for the heuristic solution with parameter vector λ and $x(\lambda)$ the corresponding solution. Then we have to find a vector λ so that $z(\lambda) \leq z(\mu)$ for all $\mu \in \mathbb{Q}^4$, that means we have to find

$$z^{pgreedy} = \min\{z(\lambda) : \lambda \in \mathbb{Q}^4\}. \tag{25}$$

According to [8] the parameter search space may be restricted to a compact subset of the unbounded set $\mathbb{Q}^4$. The theoretical basis for this step is given below. Let $p \in \mathbb{N}$ be the dimension of the parameter search space.

Theorem 1 *Let $\lambda, \lambda' \in \mathbb{Q}^p$. If there is a positive scalar $t \in \mathbb{Q}_+$ such that $\lambda' = t \cdot \lambda$ then $x(\lambda) = x(\lambda')$ and hence $z(\lambda) = z(\lambda')$.*

Corollary 1 *Let $\|.\|$ be an arbitrary norm. For every solution $x(\lambda)$ with $\lambda \in \mathbb{Q}^p \setminus \{0\}$ there is a $\lambda' \in \mathbb{Q}^p$ with $\|\lambda'\| = 1$ such that $x(\lambda) = x(\lambda')$.*

For the proofs of Theorem 1 and Corollary 1 we refer to [8] or [9]. These results lead us to

$$z^{pgreedy} = \min\{z(\lambda) : \|\lambda\| = 1\}. \tag{26}$$

This in particular means that the parameter search space is restricted to the unit hypersphere in $\mathbb{Q}^4$, which can speed up the computation of the vector λ considerably.

In order to find appropriate parameters λ a straightforward or randomized sampling over the surface of the hypersphere might be considered, but it turns out to be inefficient. Instead we apply a procedure that iteratively looks for a better parameter vector based on the location of the best point found so far. This algorithm is called *improving hit-and-run* (IHR for short). IHR was introduced by Zabinsky et al. [25] and is a randomized Monte-Carlo algorithm for global optimization problems. Note, that the bounded parameter domain is essential to apply IHR. Without Corollary 1 IHR would not be applicable.

We apply this algorithm to determine appropriate parameters for our greedy scoring function. The idea of IHR is to randomly generate a new candidate point based on the location of the current best point. It is worth mentioning, that points which are structurally nearer to the current best point have a higher hitting probability. If the new candidate point is an improvement (objective function value) it is kept and replaces the current best point. This parameter optimization technique was also used for the other applications of PGreedy. Next, a detailed description of the algorithm is given.

IHR is supposed to compute the parameters controlling the greedy scoring function and hence the selection strategy. It calls the PGreedy procedure as a black-box to yield a new objective function value $z(\lambda)$. The basic steps executed by the IHR are the following. Let $B \subset \mathbb{Q}^4$ be a bounded subset and $z : B \to \mathbb{Q}$ a given objective function. In the first iteration $k := 0$ we start with $\lambda_0 \in B$. Then the following steps are repeated until a stopping criterion is met. Generate a new candidate point w_{k+1} by randomly choosing a direction d_k uniformly distributed on the boundary of the unit hypersphere. With a random

Table 1 Characteristics of two full instances.

Instance	No. of pupils	No. of schools	No. of vehicle types
Bentheim	76	7	5
Aurich	696	53	11

selection of a distance $t_k > 0$ uniformly distributed on the feasible line segment along the direction from the current point, we receive the new candidate point $w_{k+1} := \lambda_k + t_k \cdot d_k$. If the candidate is improving, i.e. $z(w_{k+1} < z(\lambda_k)$, then $\lambda_{k+1} := w_{k+1}$, otherwise $\lambda_{k+1} := \lambda_k$. Finally, iteration number k is increased by 1.

A stopping criterion may be the iteration number reaches a certain limit or a time limit is specified as we used for our problem instances.

Local Search

A generated route is assigned a certain vehicle with the greedy best capacity. After finishing the greedy solution we have a look at how many load (passenger and wheelchair) is in fact transported and can decide for a cheaper vehicle type, if a school taxi of this kind is available.

5 Computational Results

In order to solve our heterogeneous PDPTW we address it from two sides. At first we use a Branch-and-Cut procedure to solve our given instances to global optimality. We use the presented formulation for the Branch-and-Cut method. Then we apply our PGreedy metaheuristic to produce near optimal solutions to our PDPTW instances.

We are dealing with two real-world instances from our industry partner ZIV (Zentrum für integrierte Verkehrssysteme, Darmstadt). Both are named according to the German counties Bentheim and Aurich, from where the data was drawn from. The size of the complete data set is given in table 1. Our computational experiments were carried out on a computing node with a 12-core Intel Xeon X5675 at 3.07 GHz and 48 GB RAM running Red Hat Linux Server 6.5. As solver for the MILPs we used ILOG CPLEX 12.6 [13]. Each CPLEX run uses 12 threads and the maximum computation time is 24 hours. Our heuristic is implemented in Java using JDK6. For each run only a single thread is used.

Both instances are of a size that CPLEX' Branch-and-Cut algorithm is not capable of solving. To get a detailed look at the computational limits of this type of problem instances we extracted some small and mid-sized instances from both given instances Bentheim and Aurich. Table 2 shows the size of these extracted instances together with the complete instance. The column 'setting' consists of three subcolumns 'np' for the number of pupils (or requests), 'ns' for the number of different schools and 'nv' for the number of available vehicles.

Table 2 Setting of the extracted instances

instance	np	ns	nv	instance	np	ns
ben_02	2	2	2			
ben_03	3	2	4			
ben_05	5	2	6	aur_100	100	12
ben_10	10	3	9	aur_200	200	25
ben_20	20	4	11	aur_300	300	28
ben_30	30	6	14	aur_400	400	40
ben_40	40	7	16	aur_500	500	47
ben_50	50	7	18	aur_600	600	50
ben_full	76	7	21	aur_full	696	53

Table 3 Optimal solutions of the extracted instances

	instance size			instance size preproc.			solution	
inst	var	cons	nz	var	cons	nz	opt	t
ben_02	70	138	470	35	42	149	58.72	0.01
ben_03	189	444	1,626	117	193	707	85.27	0.05
ben_05	715	2,010	7,700	440	894	3,586	165.95	0.23
ben_10	4,349	14,377	56,479	2,715	6,965	28,500	221.24	339.80
ben_20	19,411	70,293	278,831	11,690	35,154	143,714		

The Aurich instance are large-scale instances, which we only use to apply our heuristic hence no vehicle number is given and needed.

CPLEX is only capable of solving small instances. The results of the small instances extracted from the Bentheim instance are given in table 3. The size of the instances is given in the subcolumns of 'instance size' where the number of variables ('var'), the number of constraints ('cons') and the number of non-zeros ('nz') is presented respectively. The subcolumns of 'instance size preproc.' provide the size of the reduced MILP instances, that is the instance size after the preprocessing step of CPLEX. CPLEX was only capable of solving instances up to the size of 10 pupils, the associated objective function values and computing times for these instances are provided in column 'opt' and column 't' respectively. Instance ben_20 and all the other instances which have a higher number of requests are to large to be solved to proven optimality by CPLEX' Branch-and-Cut algorithm. However, for the instance ben_20 we could at least obtain an incumbent solution with an objective function value of 339.65 and a remaining gap of 42.46%.

We have used the proposed PGreedy heuristic to address our problem instances. For each of the instances we computed 10 runs and set a time limit of 300 seconds to optimize the parameters of the scoring function by improving hit-and-run. The results are given in table 4 and table 5, where the best objective function values, found within these 10 runs, are provided in column 'best'. The median value of the objective function values, which is the average of the 5th and 6th element, is shown in column 'median', the average value of all runs are given in column 'mean' and the worst objective function value can be obtained from column 'worst'. The variance of the PGreedy output is low as

Table 4 PGreedy results of the Bentheim instance and the instances extracted from it

inst	best	median	mean	worst
ben_02	58.72	58.72	58.72	58.72
ben_03	85.27	85.27	85.27	85.27
ben_05	165.95	165.95	165.95	165.95
ben_10	239.57	239.57	239.57	239.57
ben_20	357.78	357.78	357.92	361.99
ben_30	547.52	557.02	555.67	565.98
ben_40	691.21	707.17	706.82	716.19
ben_50	796.34	816.16	815.63	826.86
ben_76	1,040.20	1,061.57	1,061.18	1,083.77

Table 5 PGreedy results of the Aurich instance and the instances extracted from it

inst	best	median	mean	worst
aur_100	1,745.11	1,749.46	1,749.46	1,753.81
aur_200	2,859.13	2,865.14	2,865.14	2,871.14
aur_300	3,697.44	3,713.38	3,713.38	3,729.32
aur_400	4,426.42	4,428.69	4,428.69	4,430.96
aur_500	5,026.46	5,042.74	5,042.74	5,059.01
aur_600	5,643.41	5,665.36	5,665.36	5,687.32
aur_696	6,020.56	6,044.23	6,044.23	6,067.90

Table 6 Number of used vehicles computed by PGreedy

inst	PGreedy	Current Schedule	Saving Potential	OptiTours by ZIV
Aurich	105	130	19.2%	109
Bentheim	12	-	-	13

one can see best and worst values are very close to each other. Unfortunately we are only able to compare our heuristic values with Bentheim instances up to a number of pupil requests of 10, because all other instances are too large to yield a global optimal solution by CPLEX due to limited main memory. For the instance ben_10 we obtain a heuristic solution that is 8.3% worse than the global optimum. For the incumbent solution of instance ben_20 our heuristic solution only differs by around 5%. For the small instances heuristic solution and global optimal solutions are equal.

Until now, we used a fixed drop time of five minutes for a delivery at a school independently if there are children with wheelchairs to drop or not. In a different variant of our implementation we used a variable drop time. That means we assume a service time of three minutes to drop children with wheelchairs and one minute for children without wheelchairs. With this variable service time we are able to compare our solution with the schedule currently being used for the school taxi transport in Aurich. The results are presented in table 6. According to our PGreedy solution 105 school taxis are needed to meet all pupil requests in the Aurich instance. This solution has got a saving potential of almost 20% compared to the schedule that is currently in

Table 7 Branch-and-Cut results using best heuristic solutions as initial upper bound

inst	Primal Heuristic		CPLEX with initial solution			
	obj. value	root gap	best incumbent	best node	gap	time
ben_10	239.57	7.65%	221.24	221.24	0%	245.37s
ben_20	357.78	43.26%	319.81	203.02	36.52%	24h
ben_30	547.52	57.04%	492.74	235.19	52.27%	24h
ben_40	691.21	58.94%	691.21	283.76	58.94%	24h
ben_50	796.34	61.36%	796.34	307.68	61.36%	24h
ben_76	1040.20	63.28%	1040.20	381.95	63.28%	24h

use and deploys 130 school taxis. Unfortunately we do not know the number of vehicles currently being used for the Bentheim instance and hence are not able to compare our result. However, there is an optimization software developed by our partner ZIV with which we can compete. This optimization software is called OptiTours and is based on Genetic programming. ZIV solved the given instances by running OptiTours for several hours and reported the number of vehicles according to the solutions received by OptiTours of 109 for the Aurich instance and 13 for the Bentheim instance. In both cases our PGreedy algorithm outperforms their implementation.

Finally we provide our best heuristic solution as an initial feasible solution to the Branch-and-Cut procedure of CPLEX. On one hand, the branch-and-bound procedure may also further improve the initial solution; on the other hand, providing an initial upper bound should help speed up the branch-and-bound procedure by pruning inferior subbranches earlier. As shown in Table 7, the initial heuristic solutions are further improved in instance ben_10, ben_20, and ben_30 by around 5 to 10%. However, the heuristic solution in the larger instances cannot be improved anymore. It is also found that the lower bounds are improved. However, the gap between the best integer solution and the LP relaxation bound is still quite large, from 36.5% in instance ben_20 to 63.28% in instance ben_76. A procedure with a tighter lower bound should be used.

6 Conclusion

In this article, we addressed the problem of routing school taxi to transport children with special needs to school. This application can be described as a heterogeneous pickup and delivery problem with time windows. We presented a MILP model using multi-commodity flow formulation and Miller-Tucker-Zemlin constraints. Since exact methods such as Branch-and-Cut approaches are not able to handle large-scale instances as in our case, we used the PGreedy metaheuristic to address them. PGreedy is based on parametrizing greedy scoring functions to include more than one selection criterion during route construction. The parameters are determined automatically by the global optimization technique improving hit-and-run. Currently only small instances

up to 10 pupils can be solved to proven optimality. The commercial MILP solver cannot even obtain a feasible solution within 24 hours for instances with 20 pupils and more. The general-purposed PGreedy algorithm appears to be very effective on the small problems. It reaches solution within 10% from optimum or best-known solutions within 300 seconds computation time. For the largest real world instance Aurich with 696 pupils, we observe an improvement of around 20% compare to the current manual taxi schedule. Our PGreedy heuristic also outperforms the commercial software OptiTours specialized for the school taxi routing problem.

There are a number of directions that the current work can be extended. The LP relaxation lower bound for our current MILP model is not tight enough. More work should be done to improve the lower bounds, including model reformulation, adding more problem specific valid inequalities, and Branch-and-Price procedures. On the primal aspect, we plan to extend the current PGreedy to the framework of iterated greedy, which iteratively destruct part of the complete solution and reconstruct the partial solution using a greedy approach. As this is also a new real-world network application problem, we also plan to extract more real-world instances and make these instances available online, so that other researchers can test their algorithm on this problem.

References

1. Braca, J., Bramel, J., Posner, B., Simchi-Levi, D.: A computerized approach to the new york city school bus routing problem. IIE Transactions **29**(8), 693–702 (1997)
2. Bräysy, O., Gendreau, M.: Vehicle routing problem with time windows, part ii: Metaheuristics. Transportation Science **39**(1), 119–139 (2005)
3. Crainic, T.G., Laporte, G.: Fleet Management and Logistics. Centre for Research on Transportation. Springer (1998)
4. Desrochers, M., Lenstra, J.K., Savelsbergh, M.W.P., Soumis, F.: Vehicle routing with time windows: Optimization and approximation. Vehicle routing: Methods and Studies **16**, 65–84 (1988)
5. Dumas, Y., Desrosiers, J., Soumis, F.: The pickup and delivery problem with time windows. European Journal of Operational Research **54**, 7–22 (1991)
6. El-Sherbeny, N.A.: Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. Journal of King Saud University - Science **22**(3), 123–131 (2010)
7. Ford, L.R., Fulkerson, D.R.: Constructing Maximal Dynamic Flows from Static Flows. Operations Research **6**(3), 419–433 (1958)
8. Fügenschuh, A.: Parametrized greedy heuristics in theory and practice. In: M. Aguilera, C. Blum, A. Roli, M. Sampels (eds.) Hybrid Metaheuristics, Second International Workshop, HM 2005, Barcelona, pp. 21 – 31 (2005)
9. Fügenschuh, A.: The vehicle routing problem with coupled time windows. Central European Journal of Operations Research **14**, 157–176 (2006)
10. Fügenschuh, A., Homfeld, H., Huck, A., Martin, A., Yuan, Z.: Scheduling locomotives and car transfers in freight transport. Transportation Science **42**(4), 478 – 491 (2008)
11. Hashimoto, H., Ibaraki, T., Imahori, S., Yagiura, M.: The vehicle routing problem with flexible time windows and traveling times. Discrete Applied Mathematics **154**(16), 2271 – 2290 (2006). Discrete Algorithms and Optimization, in Honor of Professor Toshihide Ibaraki at His Retirement from Kyoto University International Symposium on Discrete Algorithms and Optimization

12. Hosny, M., Mumford, C.: New solution construction heuristics for the multiple vehicle pickup and delivery problem with time windows. MIC2009, Metaheuristic International Conference (2009)
13. IBM ILOG CPLEX: Information available at `http://www.ibm.com/software/integration/optimization/cplex/` (2013)
14. Kolks, W., Fiedler, J.: Verkehrswesen in der kommunalen Praxis Band 1. Schmidt (Erich), Berlin (1997)
15. Lawler, E., Lenstra, J., Rinnooy, A., Shmoys, D.: The Traveling Salesman Problem: a Guided Tour of Combinatorial Optimization. John Wiley and Sons, Chichester, New York (1985)
16. Lu, Q., Dessouky, M.M.: A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows. European Journal of Operational Research **175**(2), 672 – 687 (2006)
17. Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulation of traveling salesman problems. Journal of the ACM (JACM) **7**(4), 326–329 (1960)
18. Nanry, W.P., Barnes, J.W.: Solving the pickup and delivery problem with time windows using reactive tabu search. Transportation Research Part B: Methodological **34**(2), 107–121 (2000)
19. Parragh, S.N., Doerner, K.F., Hartl, R.F.: A survey on pickup and delivery problems. Journal fr Betriebswirtschaft **58**(2), 81–117 (2008)
20. Reinelt, G.: The Traveling Salesman: Computational Solutions for TSP Applications. Springer Verlag, Berlin (1994)
21. Savelsbergh, M.W.P., Sol, M.: The general pickup and delivery problem. TRANSPORTATION SCIENCE **29**, 17–29 (1995)
22. Solomon, M.M.: Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research **35**, 254–265 (1987)
23. Toth, P., Vigo, D.: The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications (2002)
24. Y. Koskosidis W. Powell, M.S.: An optimization-based heuristic for vehicle routing and scheduling with soft time windows constraints. Transportation Science **26**, 57–69 (1992)
25. Zabinsky, Z.B., Smith, R.L., Mcdonald, J.F., Edwin, H., Kaufman, D.E.: Improving hit-and-run for global optimization. Journal of Global Optimization **3**, 171–192 (1993)